BAB II

LANDASAN TEORI

2.1 Teori Umum

2.1.1 Data

Menurut Inmon (2002, p388), Data adalah sebuah rekaman dari fakta, konsep ataupun instruksi pada sebuah media peyimpanan untuk komunikasi, pengambilan maupun pemrosesan dari pengertian otomatis dan presentasi dari informasi yang dapat dimengerti oleh manusia.

Menurut Turban, Rainer dan Potter (2005, p38), data adalah deskripsi dasar tentang sesuatu, kejadian, kegiatan dan transaksi yang direkam, diklasifikasikan dan disimpan, namun tidak terorganisir untuk menyampaikan arti khusus.

Jadi, data adalah sumber informasi yang berupa rekaman dari fakta dan yang nantinya dapat digunakan menjadi suatu bentuk yang lebih berguna dan berarti.

2.1.2 Informasi

Tahapan berikutnya dari data adalah informasi, dimana berdasarkan pendapat Inmon (2002, p391), Informasi adalah kumpulan data yang telah diolah dan dievaluasi yang dapat digunakan untuk penyelesaian masalah maupun membantu dalam membuat keputusan.

Menurut Whitten (2004, p27), informasi merupakan data yang diproses atau diorganisasikan ke dalam suatu bentuk yang memiliki arti untuk seseorang. Informasi dibentuk dari berbagai kombinasi data yang diharapkan dapat memberikan makna bagi penerimanya.

Dari pendapat tersebut, maka dapat ditarik kesimpulan bahwa informasi yang diperlukan oleh perusahaan untuk pengambilan keputusan. Informasi ini adalah merupakan kumpulan dari data-data yang dimiliki oleh perusahaan, dan diolah sedemikian rupa sesuai dengan keperluannya.

2.1.3 *Database*

Menurut Inmon (2002, p388), pengertian dari *Database* adalah sebuah koleksi dari data yang disimpan dan memiliki keterkaitan (biasanya memiliki redundansi namun dalam jumlah yang terbatas dan masih terkendali) yang sesuai dengan skema. Sebuah database dapat melayani satu maupun banyak aplikasi.

Menurut Connoly dan Begg (2005, p15), *Database* adalah sebuah pembagian kumpulan data yang berelasi secara logika, dan keterangan data yang didesain untuk mendapatkan informasi yang dibutuhkan sebuah organisasi.

Menurut O'brien (2005,p211), *Database* adalah kumpulan elemen data yang terintegrasi yang berhubungan secara logikal.

2.1.4 DBMS (*Database Management System*)

Database Management System atau sering juga disebut DBMS, merupakan sebuah sistem peranti lunak yang dikembangkan untuk mengatur dan mengontrol akses dari *database*. Seperti dikutip dari tulisan Inmon (2002, p388), *Database Management System* (DBMS) adalah sebuah sistem piranti lunak berbasis computer yang digunakan untuk menjalankan dan mengatur data.

Pendapat Inmon diatas juga diperkuat oleh pendapat Kimball dan Ross (2002, p401) yang menyatakan *Database Management System* (DBMS) merupakan sebuah aplikasi computer yang bertujuan untuk menyimpan, mengambil dan

memodifikasi data dengan sebuah cara bertingkat tinggi yang terstruktur. Data yang berada didalam DBMS ini umumnya dibagi berdasarkan ragam aplikasi.

2.1.5 *Data Warehouse*

2.1.5.1 Pengertian *Data Warehouse*

Pengertian mengenai *Data warehouse* diutarakan oleh Kimball dan Ross (2002, p397) yang menyatakan *Data Warehouse* merupakan perpaduan dari data sebuah organisasi, baik dari *area staging* maupun *area* presentasi, dimana data operasional secara spesifik serta terstruktur untuk *query* dan analisis performansi dan memudahkan penggunaan.

Menurut W.H. Inmon (2002, p31), *Data Warehouse* adalah kumpulan data yang mendukung pengambilan keputusan manajemen yang memiliki karakteristik berorientasi pada subjek (*Subject Oriented*), terintegrasi (*Integrated*), mempunyai variasi waktu tertentu (*Time-Variant)* dan tidak dapat berubah (*Non-Volatile*).

Berdasarkan Turban, Rainer dan Potter (2005, p69), *Data Warehouse* adalah sebuah tempat penyimpanan data-data historis yang berorientasi subjek yang diorganisasikan agar dapat diakses dalam bentuk yang siap disajikan untuk proses analisis.

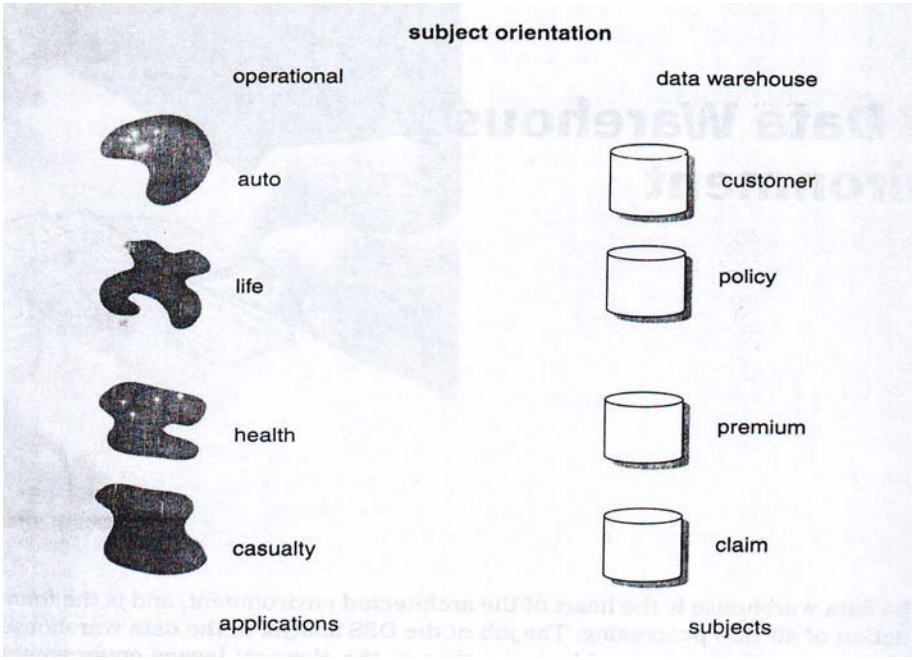
Menurut O’Brien (2005, p697), *Data warehouse* adalah kumpulan terpadi dari data yang diambil dari *database* operasional, historis, dan eksternal, yang dibersihkan, diubah, dan dikatalogkan untuk penelusuran dan analisis untuk menyediakan kecerdasan bisnis bagi pengambilan keputusan bisnis.

2.1.5.2 Karakteristik *Data Warehouse*

Seperti yang telah diutarakan sebelumnya pada sub bagian pengertian *data warehouse*, karakteristik yang harus dimiliki dalam sebuah *data warehouse* antara lain adalah *Subject-Oriented, Integrated, Time Variant,* dan *Non Volatile*. Berikut ini adalah penjelasan lebih lanjut mengenai karakteristik ini berdasarkan Inmon (2002, p31-35).

a. Subject-Oriented

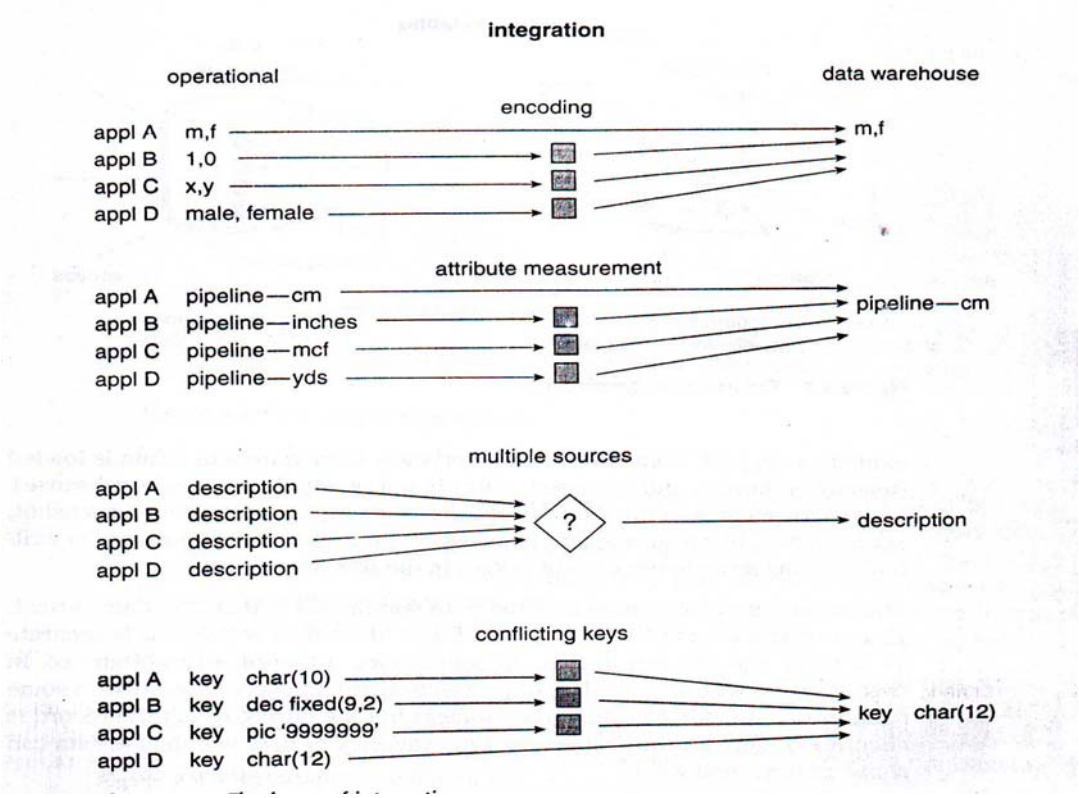
Subject-Oriented artinya data diorganisir berdasarkan topic bisnis bukan oleh nomor pelanggan, atau nomor lainnya. Orientasi subyek dalam *data warehouse* ditunjukan pada Gambar 2.1 di bawah ini. Sistem operasi klasik terorganisir seputar aplikasi dari perusahaan. Untuk perusahaan asuransi, aplikasinya adalah *auto, health, life* dan *casualty*. Area subjek utama dari perusahaan asuransi adalah *customer, policy, premium* dan *claim*.



Gambar 1 Data yang berorientasi subjek (Inmon, 2002, p32)

b. Integrated

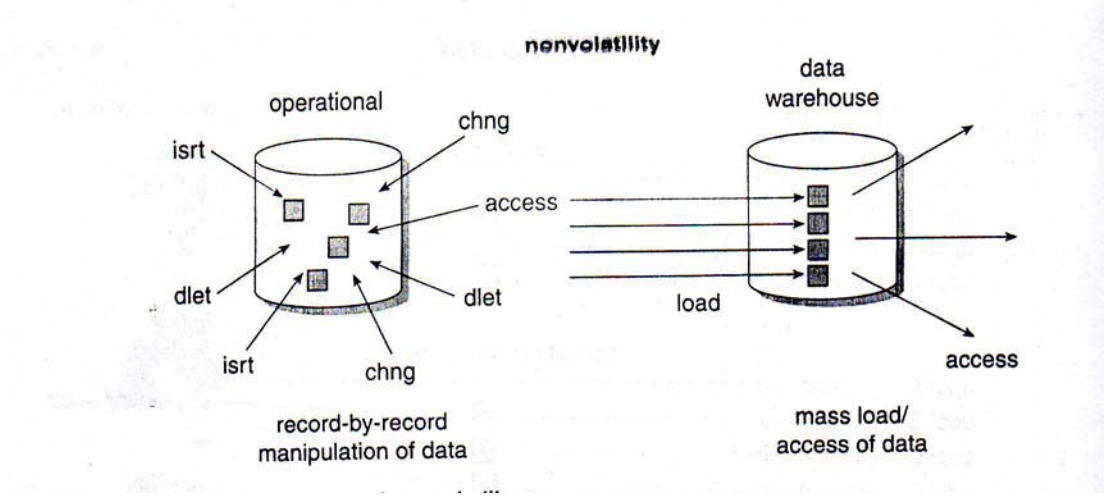
Integrated artinya data disimpan sebagai unit tunggal, bukan sebagai kumpulan *file-file* yang mungkin mempunyai struktur atau pengaturan yang berbeda. Dari semua aspek dalam *data warehouse* integrasi adalah aspek yang paling penting. Data dalam *data warehouse* diambil dari seumber beragam yang terpisah. Saat data tersebut diambil, data diubah, diformat ulang, diringkas, dirangkai ulang dan seterusnya. Hasilnya, ketika telah terletak dalam *data warehouse,* data memiliki gambaran fisik terpadu yang tunggal. Gambar 2.2 berikut mengilustrasikan integrasi yang terjadi pada saat data dari lingkungan operasional yang berorientasi aplikasi melewati *data warehouse*.



Gambar 2 Masalah mengenai integrasi (Inmon, 2002, p33)

c. Non-Volatile

Non-volatile artinya data tidak terus menerus berubah, data baru dapat ditambahkan berdasarkan jadwal tetapi data lama tidak dibuang. Sebagaimana mestinya, data dalam lingkungan operasional diperbaharui tetapi data dalam *data warehouse* menunjukkan serangkaian karakteristik yang berbeda. Data dalam *data warehouse* biasanya diisi dan diakses tetapi tidak diperbaharui. Sebaliknya, data akan dimuat dalam *snapshot static format* saat data dalam *data warehouse* dimuat. Saat perubahan berikutnya terjadi, *snapshot record* yang baru ditulis. Dengan begitu, sejarah data disimpan dalam *data warehouse*.

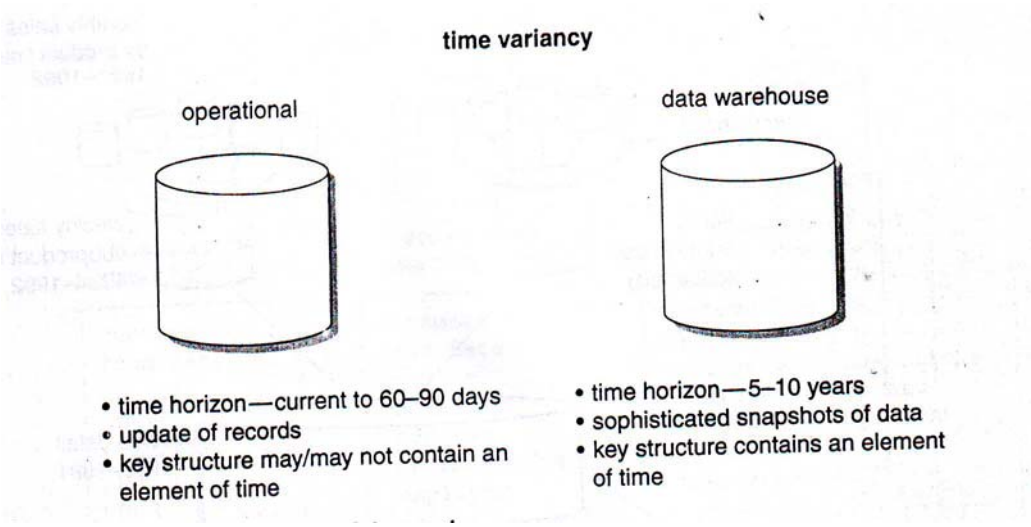


Gambar 3 Masalah mengenai *Nonvolatility* (Inmon, 2002, p34)

d. Time Variant

Karakteristik terakhir yang menonjol dari *data warehouse* adalah variasi waktu. *Time Variant* artinya dimensi waktu secara eksplisit termasuk dalam data sehingga kecenderungan dan perubahan seiring waktu dapat dipelajari. Perbedaan waktu menunjukkan bahwa setiap unit data dalam *data*

warehouse akurat satu kali dalam satu waktu. Dalam beberapa kasus, sebuah *record* diberi keterangan waktu. Dalam kasus lainnya, sebuah *record* memiliki tanggal transaksi. Tetapi pada setiap kasus, terdapat beberapa bentuk penandaan waktu untuk menunjukkan waktu pada saat *record* tertentu akurat. Gambar 2.4 mengilustrasikan bagaimana variasi waktu dari data dalam *data warehouse* dapat ditampilkan dalam beberapa cara.



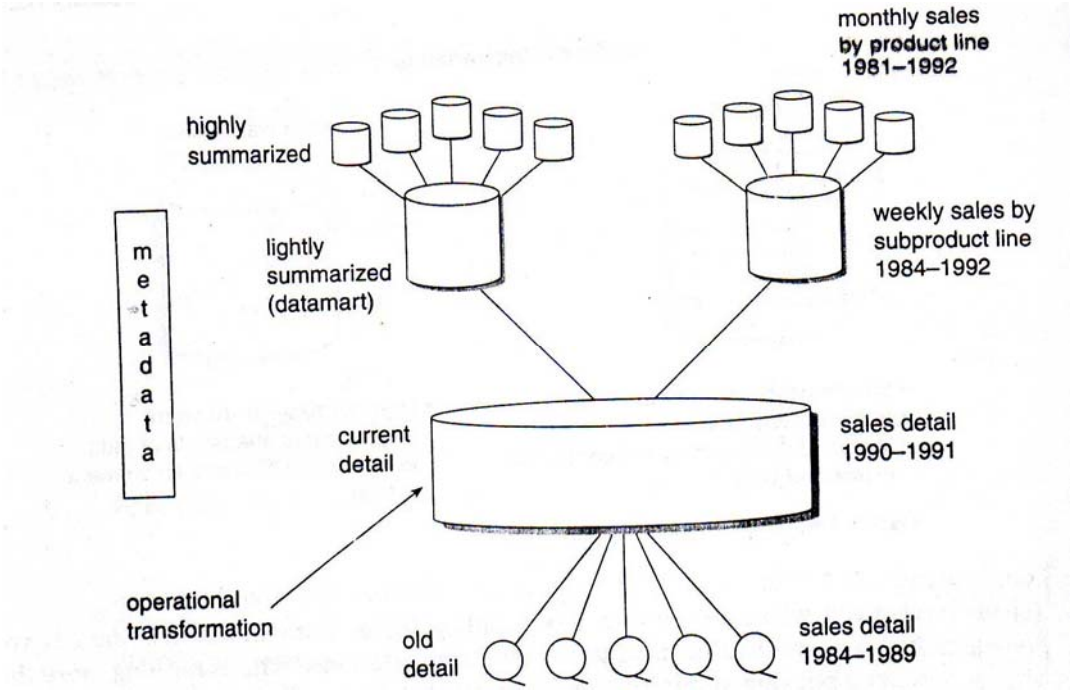
Gambar 4 Masalah mengenai *Time Variancy* (Inmon, 2002, p35)

2.1.5.3 Struktur *Data Warehouse*

Inmon (2002, p35-36) mengutarakan pendapat mengenai struktur *data warehouse* sebagai berikut Ada beberapa tingkatan detail pada lingkungan *data warehouse*. Tingkatan ini dikategorikan menjadi empat, yaitu : *Older detail level, Current detail level, Lightly summarized data level, Highly summarized data level.*

Aliran data awalnya terjadi dari lingkungan *environment* menuju *environment data warehouse*. Pada aliran data inilah proses transformasi terjadi.

Aliran data pada *data warehouse*, selanjutnya berada pada tingkatan *detail*. Seiring berjalannya waktu, data dari *Current detail level* mengalir menuju *Older detail level*. Apabila terjadi *summarize*, data akan beralih dari *Current detail level* menuju *Lightly summarized data level* yang kemudian akan menuju *Highly summarized data level.*



Gambar 5 Struktur *Data Warehouse* (Inmon, 2002, p36)

1. Current detail data

Current Detail data adalah data yang dapat diperbaharui pada suatu waktu tertentu sehingga keakuratan datanya sah. Contohnya rincian penjualan dari tahun 1990-1991.

2. Older detail data

Saat data sudah berumur lama maka data akan berpindah dari *Current Detail data ke Older Detail data* biasanya menggunakan media

penyimpanan alternative atau disebut juga *bulk storage*. Contohnya rincian penjualan dari tahun 1984-1989.

3. Lightly summarized data

Lightly summarized data adalah data rinci yang telah diringkas tetapi data ini belum dapat menjadi dasar pengambilan keputusan manajerial sebab sifatnya belum sepenuhnya ringkasan akhir. Contohnya laporan penjualan per minggu berdasarkan subproduk dari tahun 1984-1992. Kapasitas dari *lightly summarized data* lebih sedikit daripada data rinci yang ada karena adanya perbedaan level detail yang dapat diakses.

4. Highly summarized data

Highly summarized data adalah data yang diringkas dari *Lightly summarized data* yang sifatnya sudah merupakan ringkasan secara keseluruhan. Data ini memiliki tingkat granularity yang tinggi dalam *data warehouse* yang dapat membantu DSS (*Decision Support System*) *analyst* dan *user* untuk mendefinisikan dan menemukan informasi yang digunakan untuk pengambilan keputusan perusahaan. Contohnya Laporan Penjualan per bulan berdasarkan produk dari tahun 1981-1992.

5. *Metadata*

Menurut W.H Inmon (2002, p113), Komponen yang paling penting dalam *data warehouse* adalah *metadata* atau data tentang data, telah menjadi bagian dari lingkungan pergaulan proses informasi sejak adanya program dan data. Akan tetapi, metadata dalam dunia data warehouse membawa kepada tingkat kepentingan yang baru yaitu memberikan kegunaan yang

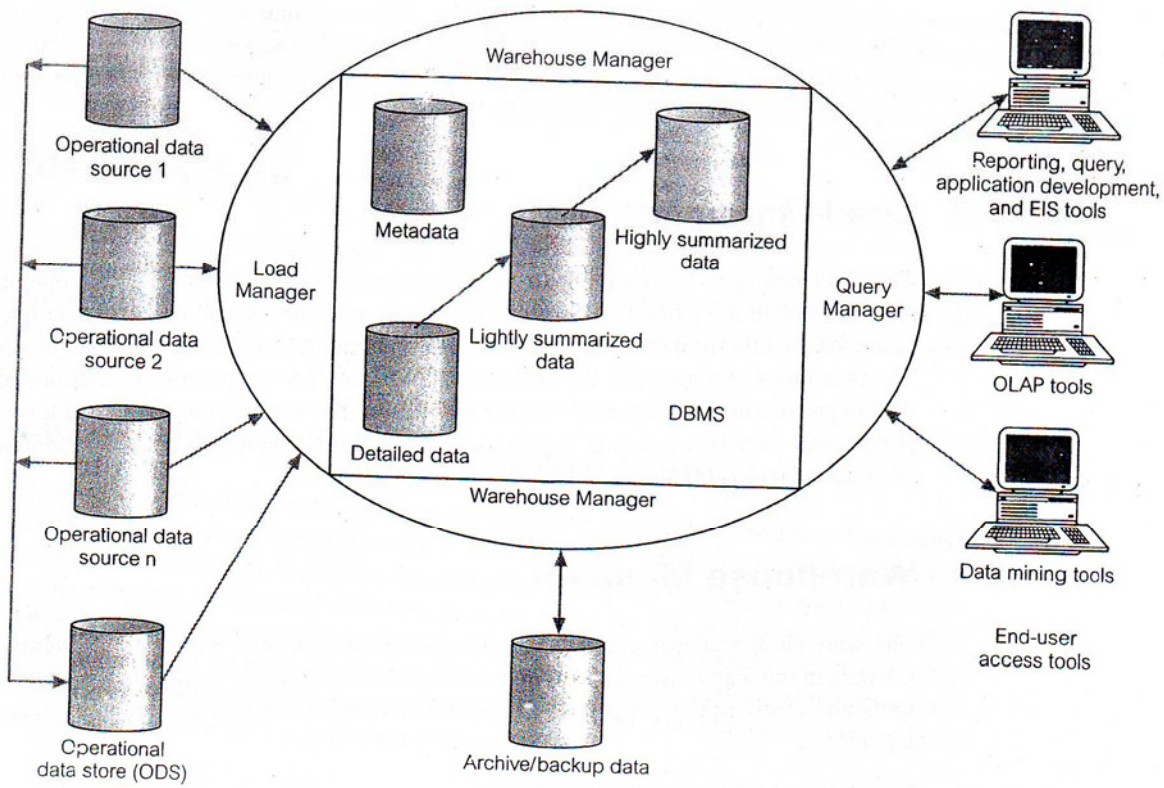
paling efektif dari *data warehouse*. Metadata mengijinkan pengguna/DSS *analyst* untuk melakukan navigasi melalui berbagai kemungkinan.

Metadata digunakan untuk tujuan yang beragam termasuk berikut ini Connoly dan Begg (2005, p1159) :

1. Proses *loading* dan *extraction – metadata* digunakan untuk memetakan sumber data ke dalam gambaran yang umum dari data dalam *data warehouse*.
2. Proses pengelolaan *warehouse – metadata* digunakan untuk mengotomatisasi produksi table ringkasan.
3. Bagian dari proses pengelolaan *query – metadata* digunakan untuk mengarahkan *query* pada sumber data yang paling tepat.

2.1.5.4 Arsitektur *Data Warehouse*

Dalam melakukan perancangan *data warehouse*, harus ditentukan terlebih dahulu arsitektur yang paling cocok untuk melakukan pengembangan *data warehouse.* Menurut Connoly dan Begg (2005, p1156) memberikan suatu gambaran *typical* arsitektur dari *data warehouse*, yaitu sebagai berikut :



Gambar 6 *Typical architecture of a data warehouse* (Connoly dan Begg, 2005, p1157)

Komponen-komponen yang ada dalam arsitektur ini dijelaskan lebih lanjut oleh Connoly dan Begg (2005, p1157-) sebagai berikut :

1. Operational Data

Sumber data untuk *Data Warehouse* disediakan dari :

- *Mainframe* data operasional ditempatkan di dalam generasi pertama hirarki dan database jaringan.
- Data *departmental* ditempatkan di dalam sistem file proprietary seperti VSAM, RMS dan *Relational* DBMS seperti Informix dan Oracle.
- Data privasi ditempatkan di dalam workstations dan server pribadi.

- Sistem External seperti Internet, database yang bersifat perdagangan tersedia, database berhubungan dengan *supplier* atau pelanggan perusahaan.

2. *Operational Data Source* (ODS)

ODS merupakan tempat penyimpanan data operasional terkini dan terintegrasi yang digunakan untuk analisis. Seringkali mempunyai struktur dan data seperti *data warehouse*, namun pada kenyataannya hanya bertindak sebagai *staging area* bagi data untuk dipindahkan ke dalam *data warehouse*.

3. *Load Manager*

Load manager yang juga dikenal dengan sebutan *frontend component*, melakukan semua operasi yang berhubungan dengan *extraction*, dan *loading* data ke dalam *data warehouse*. Selain itu, operasi yang biasanya juga ada dalam *load manager* ini adalah *transformation* sederhana untuk mempersiapkan data sebelum masuk ke *data warehouse*.

4. *Warehouse Manager*

Warehouse Manager melakukan semua operasi yang berkaitan dengan *management* dari *data warehouse.* Beberapa operasi yang dilakukan oleh *warehouse manager* antara lain adalah :

- Menganalisis data untuk memastikan konsistensi
- Mentransformasi dan menggabungkan berbagai sumber data
- Membuat *index* dan *view* pada *base tables*
- Melakukan denormalisasi
- Melakukan *aggregation*

- *Backup* dan *archiving data.*

5. *Query Manager*

Query Manager yang juga dikenal dengan sebutan *backend component*, melakukan semua operasi yang berhubungan dengan *management* dari *query* pengguna. Termasuk juga mengarahkan *query* ke table yang bersangkutan dan menjadwalkan eksekusi *query*.

6. *Detailed Data*

Area ini pada *data warehouse*, menyimpan semua data-data *detail* yang berasal dari *database source*.

7. *Lightly and Highly Summarized Data*

Area ini menyimpan semua *predefine lightly* dan *highly summarized data* yang di-*generate* oleh *warehouse manager*.

8. *Archive / Backup Data*

Area ini menyimpan semua data-data *detail* dan *summarized*, untuk tujuan *backup*. Data ditransfer ke *storage archives* seperti *magnetic tape*, atau *optical disk*.

9. Metadata

Area *warehouse* ini menyimpan semua *metadata* (Data tentang data) definisi dipakai untuk semua proses di *warehouse*. *Metadata* dipakai untuk beberapa tujuan, diantaranya :

- Proses *loading* dan *extraction – metadata* digunakan untuk memetakan sumber data ke dalam gambaran yang umum dari data dalam *data warehouse*.

- Proses pengelolaan *warehouse* – *metadata* digunakan untuk mengotomatisasi produksi table ringkasan.
- Bagian dari proses pengelolaan *query* – *metadata* digunakan untuk mengarahkan *query* pada sumber data yang paling tepat.

10. End User Access Tools

Tujuan pokok dari pembuatan *data warehouse* adalah untuk menyediakan informasi kepada pelaku bisnis untuk membuat keputusan yang strategis. Oleh karena itu diperlukan alat yang dapat menghubungkan *user* dengan *data warehouse*. Ada lima kategori alat yang dapat digunakan :

- *Reporting and query tools*
- *Application development tools*
- *Executive information system* (EIS) *tool*
- *Online analytical processing* (OLAP) *tool*
- *Data mining tool*.

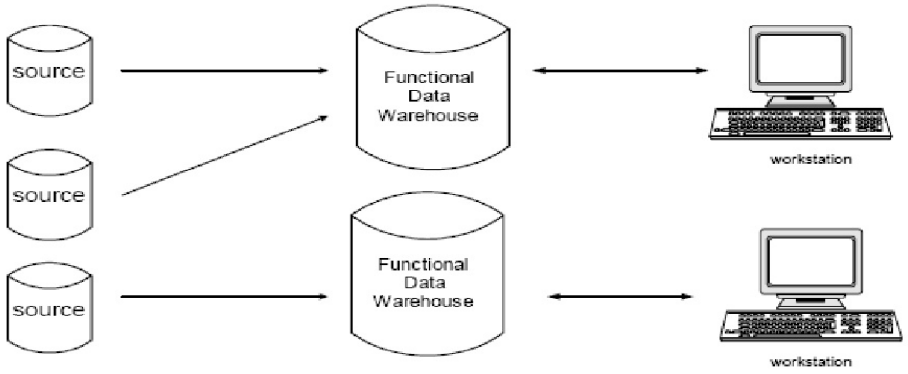
2.1.5.5 Anatomi *Data Warehouse*

a. *Data Warehouse* Fungsional

Tiap *data warehouse* Fungsional mencakup sebuah grup tersendiri yang terpisah (seperti divisi), area fungsional, unit geografis, atau grup pemasaran produk. (*http://www.etfinancial.com/dataglossary.htm*)

Data Warehouse fungsional berfokus pada kebutuhan dari sebuah fungsi bisnis, misalkan departemen, divisi dan sebagainya. Keuntungan dari *data warehouse* ini adalah memberikan fleksibilitas karena dapat disesuaikan dengan permasalahan bisnis spesifik dan kemungkinan dari departemen atau lini bisnis

tertentu, disamping relatif lebih murah dan lebih sederhana untuk diimplementasikan. Perusahaan umumnya membangun beberapa rangkaian *data warehouse* fungsional untuk mendukung area yang berbeda-beda, dan hal ini memberikan pengembangan yang cepat. Perusahaan juga dapat memberikan respon yang lebih cepat terhadap kesempatan pasar. Namun, terdapat resiko hilangnya konsistendi data di luar lingkungan fungsi bisnis bersangkutan. Apabila pendekatan ini lingkupnya diperbesar dari lingkungan fungsional menjadi lingkup perusahaan, konsistensi data perusahaan tidak dapat dijamin. (*www.freesoft.hu/download.emt?id=76*)



Gambar 7 Bentuk *Data Warehouse* Fungsional
(*http://www.scribd.com/doc/38739124/Data-Warehouse*)

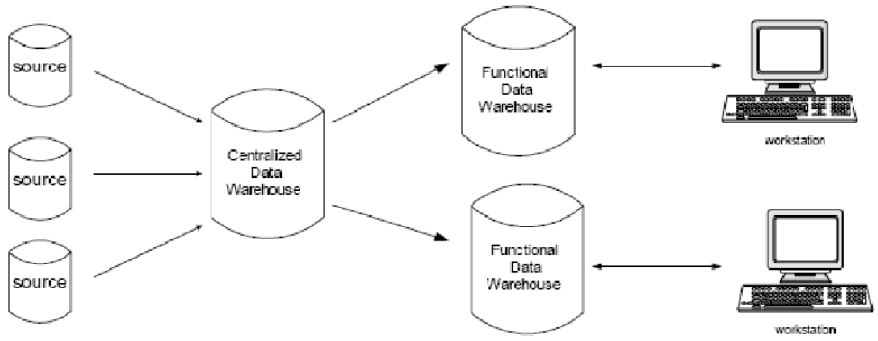
b. Data Warehouse Terpusat

Data Warehouse terpusat adalah sebuah *database* yang diciptakan dari pengekstraksian operasional yang menganut pada sebuah model data tunggal *enterprise* yang konsisten untuk memastikan konsistensi atas data pendukung keputusan dalam perusahaan. Merupakan penerapan gaya komputerisasi dimana

semua sistem informasi dilokasikan dan dimanajemen dari sebuah lokasi fisikal tunggal. (*http://www.etfinancial.com/dataglossary.htm*)

Data warehouse terpusat adalah sebuah database fisikal tunggal yang menyimpan semua data untuk are fungsional spesifik, departemen, divisi atau perusahaan (*enterprise*). Pendekatan ini umumnya digunakan saat terdapat banyak *end-user* yang sudah terhubung dengan sebuah komputer atau jaringan pusat. *Data warehouse* terpusat biasanya menyimpan data dari sistem operasi yang berbeda-beda. Data yang disimpan didalamnya dapat diakses dari sebuah lokasi dan harus di-*load* dan dipelihara pada basis data regular. (*http://www.kenorrinst.com/pg%2033%20d.w.%20whitepaper.htm*)

Data warehouse terpusat melingkupi sebuah *data warehouse* tunggal yang melayani semua kebutuhan perusahaan. Tujuan dari pendekatan ini adalah untuk memecahkan permasalahan organisasional yang membatasi operasi perusahaan. Jadi, membangun sebuah *data warehouse* terpusat yang terunifikasi sangat kompleks, membutuhkan biaya besar dan waktu lebih banyak. Namun, keuntungan dari *data warehouse* terpusat adalah menyediakan gambaran yang komprehensif, tingkat control dan reliabilitas yang tinggi karena keterpaduan data didalamnya. *(http://www.freesoft.hu/download.cmt?id=76*)



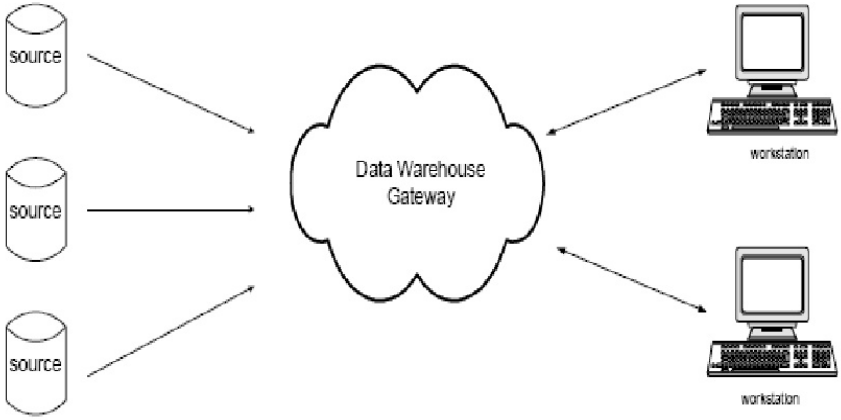
Gambar 8 Bentuk *Data Warehouse* Terpusat
(http://www.scribd.com/doc/38739124/Data-Warehouse)

c. *Data Warehouse* Terdistribusi

Data warehouse terdistribusi adalah sebuah sumber data terpisah yang dapat diakses *user via gateway* pusat yang menyediakan *view logical* atas data *corporate* dalam gambaran yang dapat dipahami oleh *user*. *Gateway* tersebut akan melakukan *parse* dan mendistribusikan *query* secara *real-time* ke sumber data terpisah dan mengembalikan *result set*-nya ke *user*. (*http://www.etfinancial.com/dataglossary.htm*)

Data warehouse terdistribusi adalah *data warehouse* yang komponennya didistribusikan ke beberapa *database* fisikal yang berbeda. Pendekatan ini umumnya dipilih saat perusahaan besar ingin mengikutsertakan level organisasinya yang lebih rendah dalam pengambilan keputusan, sehingga diperlukan penurunan data untuk pembuatan keputusan ke komputer lokal tempat pengambil keputusan lokal. Umumnya, *data warehouse* terdistribusi melibatkan data yang paling redundan dan konsekuensinya adalah proses *load and update* yang sangat kompleks. Pendekatan ini memerlukan biaya yang sangat besar

karena setiap sistem pengumpul data fungsional dan sistem operasinya dikelola secara terpisah. Disamping itu, supaya berguna bagi perusahaan, data harus disinkronisasikan untuk memelihara keterpaduannya. (*http://www.kenorrinst.com/pg%2033%20d.w.%20whitepaper.htm*)



Gambar 9 Bentuk *Data Warehouse* Terdistribusi
(http://www.scribd.com/doc/38739124/Data-Warehouse)

2.1.5.6 Kegunaan *Data Warehouse*

Menurut Wikipedia (*http://en.wikipedia.org/wiki/Data_warehouse*), berikut kegunaan *data warehouse* **:**

1. Pembuatan Laporan

   Pembuatan laporan adalah salah satu kegunaan data warehouse yang paling umum. Dengan menggunakan *query-query* sederhana dalam *data warehouse*, dapat dihasilkan informasi per tahun, per kuartal, per bulan, atau bahkan per hari. *Query-query* tersebut digunakan dengan tujuan memperoleh jawaban atas pertanyaan-pertanyaan khusus, seperti kapan, siapa, dimana, dan sebagainya.

2. OLAP (*Online Analytical Processing*)

   OLAP adalah proses per bagian untuk lingkungan *data mart*. *Data warehouse* digunakan dalam melakukan analisa bisnis untuk menyelidiki kecenderungan pasar dan faktor-faktor penyebabnya karena dengan adanya *data warehouse*, semua informasi baik rincian maupun ringkasan yang dibutuhkan dalam proses analisa mudah didapat. Dalam hal ini, *data warehouse* merupakan *tool* yang handal untuk analisa data yang kompleks.

3. Data Mining

   Penggunaan *data warehouse* dalam pencarian pola dan hubungan data, dengan tujuan membuat keputusan bisnis bagi para pihak manajemen. Dalam hal ini, *software* dirancang untuk pola statistic dalam data untuk mengetahui kecenderungan yang ada, misalnya kecenderungan pasar akan suatu produk tertentu.

4. Proses informasi eksekutif

   Data warehouse untuk mencari ringkasan informasi yang penting dengan tujuan membuat keputusan bisnis tanpa harus menjelajahi keseluruhan data. Dengan menggunakan *data warehouse,* segala laporan telah diringkas, dan dapat pula diketahui rinciannya secara lengkap, sehingga mempermudah proses pengambilan keputusan, dan data pada laporan *data warehouse* menjadi sangat informatif bagi *user,* dalam hal ini pihak eksekutif.

2.1.5.7 Metodologi Perancangan *Data Warehouse*

2.1.5.7.1 Konsep Pemodelan *Database*

Terdapat dua konsep pemodelan *database* yang umum digunakan, yaitu ER (*Entity-Relation*) *Modelling* dan *Dimensionality Modelling* berdasarkan keterangan yang diutarakan oleh Conolly dan Begg (2005, p1182). *Dimensionality Modelling* lebih digunakaan dalam perancangan komponen *data warehouse*, sedangkan *ER-Modelling* lebih umum digunakan pada pemodelan *database* OLTP.

2.1.5.7.1.1 *Entity Relationship* (ER) *Modelling*

Berdasarkan keterangan dari Connolly dan Begg (2005, p1186), ER *Modelling* merupakan teknik yang mengidentifikasi relasi antar entitas, tujuan utama dari ER *modeling* adalah menghilangkan redundansi pada data, model ini biasa digunakan pada perancangan *database* untuk OLTP.

2.1.5.7.1.2 *Dimensionality Modelling*

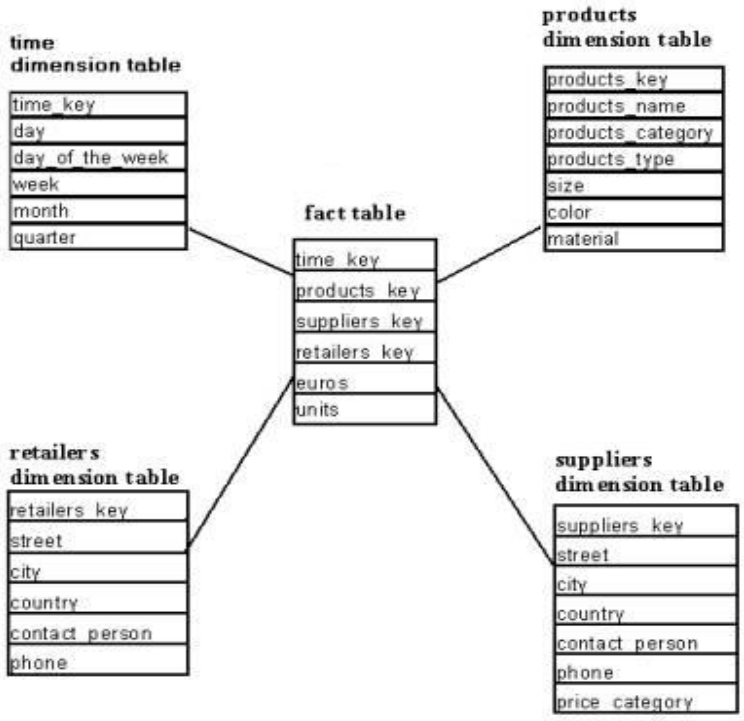
Menurut Connolly dan Begg (2005, p1183), *dimensionality modeling* adalah sebuah teknik *logical design* yang bertujuan untuk menghadirkan data dalam sebuah bentuk yang standard dan intuitif yang memungkinkan pengaksesan *database* dengan performansi yang tinggi.

Ada beberapa konsep pemodelan *data warehouse* pada *dimensionality modeling* yang dikenal umum pada saat ini yaitu antara lain adalah *Star Schema, Snowflake Schema,* dan *Starflake Schema.*

1. ***Star Schema*** **(Skema Bintang)**

Skema bintang merupakan salah satu konsep pemodelan *data warehouse*. menurut Connolly dan Begg (2005, p1183), Skema bintang adalah struktur logika yang memiliki table fakta yang mengandung data factual pada pusatnya dan dikelilingi oleh table dimensi yang mengandung data referensi (yang dapat didenormalisasi).

Skema bintang merupakan table fakta yang terdapat di tengah, yang terhubung pada serangkaian table dimensi. Skema bintang mengeksploitasi karakteristik dari data factual seperti fakta yang dihasilkan oleh kejadian pada masa lampau, dan tak mungkin untuk berubah, tanpa mempedulikan bagaimana mereka dianalisis. Contoh gambar Skema Bintang :

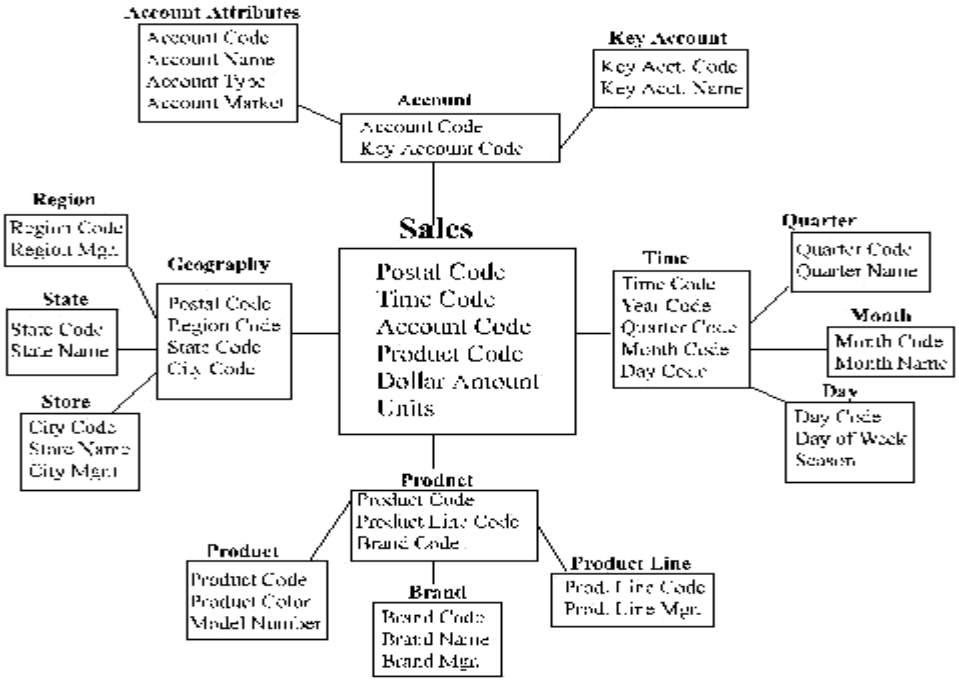


Gambar 10 Contoh Skema Bintang

(*http://blog-mstechnology.blogspot.com/2010/06/bi-dimensional-model-star-schema.html*)

2. *Snowflake Schema*

Connolly dan Begg (2005, p1184) berpendapat *Snowflake schema* adalah variasi bentuk dari skema bintang dimana pada table dimensi tidak mengandung data yang telah di-denormalisasi. Pendapat ini juga dikemukakan oleh Kimball dan Ross (2002, p413) yang menuturkan *Snowflake* adalah sebuah dimensi yang telah didenormalisasi dimana sebuah table dimensi yang *flat*, dan *single* disusun ulang dengan membentuk struktur pohon yang berpotensi memiliki banyak *nesting level.* Keterangan lebih lanjut juga dinyatakan oleh Kimball dan Ross (2002, p413) bahwa pada model dimensional, table fakta tidak memiliki perbedaan, baik dengan konsep pemodelan menggunakan Skema bintang maupun *Snowflake,* tetapi tidak sama halnya dengan table dimensi, pada konsep pemodelan *snowflake* table dimensi biasanya ditampilkan dalam bentuk *third normal form*. Contoh gambar *Snowflake* :



Gambar 11 Contoh *Snowflake schema*
(*http://www.informix.com.ua/articles/rolap/rolap.htm*)

2.1.5.7.1.2.1 Komponen *Dimensionality Modeling*

a. Fact

Fact atau fakta seperti dikutip dari Kimball dan Ross (2002, p402) adalah sebuah ukuran dari performansi bisnis, biasanya berupa *numerical* dan penjumlahan. Hal ini berlanjut pada pengertian dari table *fact* sebagai lokasi penyimpanan untuk *fact* yang ada.

b. Fact Table

Ada beberapa pendapat yang mengungkapkan tentang pengertian dari *fact table* atau table fakta diantaranya adalah pendapat dari Inmon (2002, p391) yang menyatakan Tabel Fakta adalah pusat dari table *star join* dimana data dengan banyak kepentingan tersimpan. Dan pendapat mengenai pengertian *fact table* juga diutarakan oleh Kimball dan Ross (2002, p402),

yang menyatakan bahwa Tabel fakta ada sebuah skema bintang (*dimensional model*) adalah table *central* dengan pengukuran performansi bisnis dalam bentuk numeric yang memiliki karakteristik berupa sebuah *composite key*, yang tiap-tiap elemennya adalah *foreign key* yang didapat dari table dimensi.

c. ***Dimension***

Dimension atau biasa disebut dimensi merupakan sebuah entitas *independent* pada sebuah model *dimensional* yang berfungsi sebagai pintu masuk atau berperan sebagai sebuah mekanisme untuk memecah-mecah pengukuran tambahan yang ada pada table fakta dari model *dimensional*. Pendapat ini dinyatakan oleh Kimball dan Ross (2002, p399).

d. ***Dimension Table***

Dimension table atau tabel dimensi menurut Inmon (2002, p389) merupakan Tempat dimana data tambahan yang berhubungan dengan tabel fakta ditempatkan pada sebuah tabel *multidimensional*.

Pendapat lain dikemukakan oleh Kimball dan Ross (2002, p399) yang menyatakan bahwa Sebuah tabel dimensi adalah sebuah tabel pada model dimensional yang memiliki sebuah *primary key* tunggal dan kolom dengan atribut deskriptif.

e. Surrogate Key

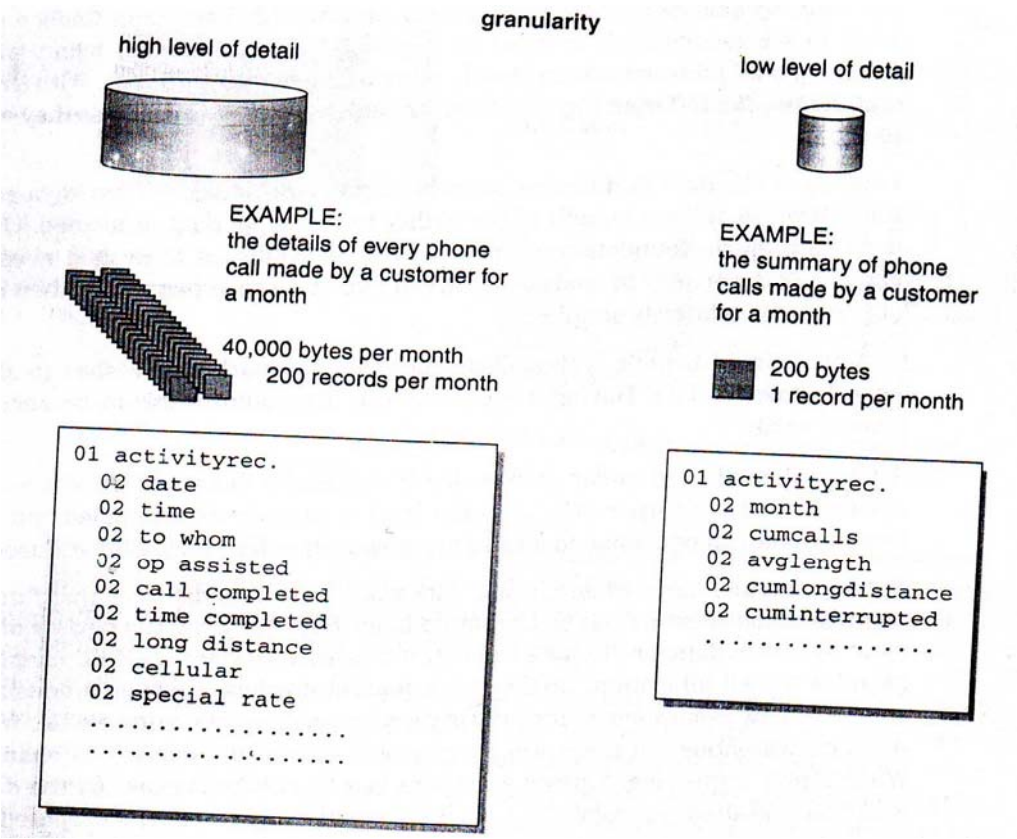
Surrogate Key merupakan salah satu elemen yang biasanya ditambahkan pada tabel dalam konsep pemodelan *data warehouse*, menurut Kimball dan Ross (2002, p414) *Surrogate Key* adalah *key* berupa *integer* yang secara *sequential* ditambahkan sesuai keperluan pada *staging area* untuk

membentuk sebuah tabel dimensi dan elemen yang menggabungkannya dengan tabel fakta. Pada tabel dimensi, *surrogate key* ini bertindak sebagai *primary key*. Sedangkan pada tabel fakta, *surrogate key* bertindak sebagai *foreign key* yang menspesifikasikan dimensi, walaupun terkadang tidak dibutuhkan *surrogate key* pada tabel fakta juga dapat bertindak sebagai bagian dari *primary key* yang dimiliki oleh table fakta. *Surrogate key* biasanya tidak bisa dijelaskan sendiri hanya melalui nilai yang terkandung didalamnya. *Surrogate key* pada *data warehouse* dibutuhkan untuk menangani permasalahan yang timbul dari *slowly changing dimensions* serta data yang hilang ataupun data yang tidak bisa digunakan.

f. Granularity

Granularity adalah tingkat detail yang ada pada *data warehouse*, pendapat ini dikemukakan oleh Kimball dan Ross (2002, p403), dan lebih diperjelas oleh Inmon (2002, p391) yang menyatakan bahwa *Granularity* adalah tingkat detail yang terkandung pada setiap unit data. Semakin detail tingkat datanya, maka semakin rendah tingkat *granularity*-nya. Dan sebaliknya, semakin rendah tingkat *detail* datanya, maka semakin tinggi tingkat *granularity*-nya.

Contoh *Granularity* dari data dapat dilihat pada gambar di bawah ini :



Gambar 12 Contoh dari *Data Granularity* (Inmon, 2002, p46)

2.1.5.7.1.2.3 Metodologi pembangunan *Data Warehouse*

Berdasarkan pernyataan dari Kimball dan Ross, serta yang dikutip oleh Connoly dan Begg (2005, p1187-1193), terdapat Sembilan tahapan dalam membangun *data warehouse* yang dikenal dengan *nine-step methodology*, yaitu :

1. **Memilih Proses (*Choosing The Process*)**

   Proses (fungsi) bisnis merujuk pada subjek masalah atau kebutuhan bisnis dan pemahaman mengenai data yang tersedia pada perusahaan. *Data warehouse* yang akan dibangun harus sesuai anggaran dan dapat menjawab masalah-masalah bisnis yang penting.

2. **Memilih Grain (*Choosing The Grain*)**

   Memilih *Grain* berarti menentukan hal yang sebenarnya dihadirkan oleh table fakta. Setelah menentukan grain-grain pada table fakta, dimensi-dimensi untuk setiap fakta dapat diidentifikasi. Pada proses ini juga tingkat *granularity* dari data akan ditentukan.

3. **Identifikasi dan membuat dimensi yang sesuai (*Identifying and Conforming the dimensions*)**

   Mengidentifikasi dimensi disertai deskripsi *detail* yang secukupnya. Ketika table dimensi berada pada dua atau lebih *data warehouse,* maka table dimensi tersebut harus mempunyai dimensi yang sama atau salah satu merupakan *subset* dari yang lainnya. Jika suatu table dimensi digunakan oleh lebih dari satu *data warehouse,* maka dimensinya harus disesuaikan.

4. **Memilih Fakta (*Choosing The Facts*)**

   Memilih fakta yang akan digunakan dalam table fakta berdasarkan proses bisnis dan *grain* yang telah ditentukan. Untuk memilih fakta perlu diketahui informasi apa saja yang dibutuhkan oleh pengguna dalam kaitannya dengan proses bisnis tertentu.

5. **Menentukan data pre-kalkulasi dari tabel Fakta (*Storing pre-calculation in the fact table*)**

   Setelah menentukan fakta, maka setiap fakta perlu diuji apakah terdapat fakta lain yang merupakan hasil kalkulasi dari fakta-fakta yang telah ditentukan. Fakta hasil kalkulasi sebaiknya disimpan di dalam table fakta, karena fakta ini akan dapat meningkatkan performansi dalam memberikan hasil *query*. Di

samping itu juga perlu diketahui bahwa dengan menyimpan fakta hasil kalkulasi dalam table fakta berarti ada tambahan penggunaan kapasitas dalam basis data.

6. **Melengkapi table dimensi (*Rounding out the dimension Tabels*)**

   Pada tahap ini, hal yang dilakukan adalah menambahkan informasi deskriptif yang berhubungan dengan setiap table dimensi. Disamping itu, untuk melengkapi atribut table dimensi dengan tepat maka perlu dilakukan identifikasi mengenai bagaimana tabel-tabel dimensi saling berhubungan.

7. **Memilih durasi dari basis data (*Choosing the duration of the database*)**

   Menentukan durasi data yang akan dimasukkan ke dalam *data warehouse* berdasarkan kebutuhan perusahaan. Hal ini perlu dilakukan supaya data yang akan dianalisis berdasarkan jangka waktu tertentu berada dalam *data warehouse*.

8. **Melacak dimensi yang berubah secara perlahan (*Tracking slowly changing dimension)***

   Slowly changing dimension dapat menjadi sebuah masalah. Ada tiga tipe dasar dari *slowly changing dimension,* yakni :

   a) Perubahan atribut dimensi yang ditulis ulang (*overwrite*)
   b) Perubahan atribut dimensi yang mengakibatkan pembuatan suatu *record* dimensi baru.
   c) Perubahan atribut dimensi yang mengakibatkan sebuah atribut alternative dibuat, sehingga kedua atribut tersebut yakni atribut yang lama dan yang baru dapat diakses secara bersamaan dalam sebuah dimensi yang sama.

9. Menentukan prioritas dan *mode* dari *query* (*Deciding the query priorities and the query modes*)

Pada tahap ini persoalan yang harus dipertimbangkan adalah membuat desain fisik yang bertujuan untuk senantiasa menjaga dan meningkatkan performansi dari *data warehouse*. Hal yang paling menjadi sorotan pada tahapan ini adalah persepsi *user* tentang permasalahan performansi yang berkaitan dengan keberadaan aggregasi dan *pre-stored summaries*.

2.1.5.8 Istilah dalam *Data Warehouse*

2.1.5.8.1 OLAP (*Online Analysis Processing*)

OLAP (Online Analysis Processing) menurut pendapat Kimball dan Ross (2002, p408) didefinisikan sebagai Kumpulan aturan yang menyediakan sebuah kerangka dimensional untuk mendukung keputusan (*Decision support*).

2.1.5.8.2 *Data Mart*

Menurut Kimball (2002, p396), *Data mart* adalah Bagian dari logikal dan fisikal dari area cakupan yang dimiliki oleh *data warehouse*. Sedangkat menurut Connoly dan Begg (2005, p1171), *Data mart* adalah merupakan bagian dari *data warehouse*, yang mendukung kebutuhan informasi bagian departemen atau fungsi bisnis tertentu.

Berikut karakteristik yang membedakan antara *data mart* dengan *data warehouse* :

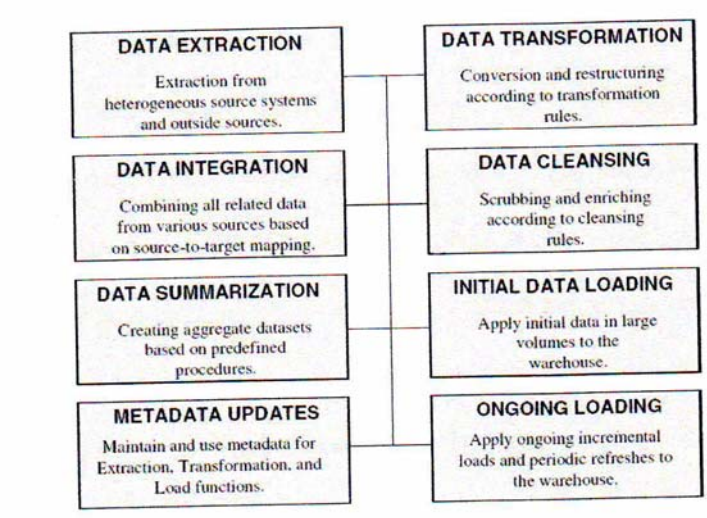
1. *Data mart* berfokus pada kebutuhan pengguna yang berhubungan dengan satu bagian departemen atau fungsi bisnis.
2. *Data mart* tidak berisi data operasional yang bersifat detil.

3. *Data mart* lebih dimengerti dan digunakan karena berisi data yang lebih sedikit dibandingkan dengan *data warehouse*.

2.1.5.8.3 ETL (*Extract, Transform and Load*)

Menurut Inmon (2002, p390), *Extract / Transform / Load* (ETL) merupakan proses yang melakukan pencarian data, mengintegrasikannya, dan menempatkannya pada sebuah *data warehouse*. Mengenai proses ETL ini, Kimball dan Ross (2002, p401) lebih menjelaskan bahwa ETL merupakan kumpulan proses yang menyiapkan data dari *operational source* untuk *data warehouse*. ETL merupakan proses utama yang terjadi di belakang *staging area* pada *data warehouse*, lebih dikedepankan dari presentasi atau proses *query* apapun. Proses ini terdiri dari proses *extracting*, *transforming*, *loading* dan melakukan beberapa proses tambahan sebelum mempublikasikannya ke *data warehouse*.

Secara garis besar, Ponniah (2001, p287) menggambarkan proses yang terjadi pada ETL pada gambar berikut ini :



Gambar 13 ETL *Summary* (Ponniah, 2001, p287)

1. *Extract*

Extraction adalah proses ekstraksi data dari *source system* untuk pemakaian lebih lanjut pada *data warehouse*. Pendapat ini dikemukakan oleh Inmon (2002, p391) yang menyatakan bahwa *Extract* adalah proses memilih data dari satu *environment* dan memindahkannya ke *environment* lain. Pada proses ETL *data warehouse, extract* merupakan langkah pertama dimana proses ini berperan untuk memilih data dari *source system* dan kemudian memindahkannya ke lokasi yang dituju yaitu *staging area* sebagai tempat proses ETL selanjutnya berjalan.

Source System yang dibicarakan disini adalah *source system* untuk *data warehouse,* yang umumnya merupakan *database* dari aplikasi OLTP. Hal yang perlu diperhatikan pada proses ekstraksi ini adalah waktu, dan tingkat kompleksitas data yang ada pada *source systems*.

2. Transform

Transform pada proses ETL *data warehouse* dimaksudkan untuk memilih, merapikan, dan memberikan atribut tambahan agar data yang telah melalui proses ekstraksi sebelumnya dapat masuk dan cocok pada skema *data warehouse* yang telah ada. Adapun hal-hal yang dilakukan pada proses transform ini antara lain adalah :

1. *Cleansing,* dilakukan guna menjaga konsistensi dari data terutama mengingat data pada *data warehouse* berasal dari berbagai sumber. Proses ini dilakukan guna memastikan integrity dari masing-masing table. Contohnya : memberikan nilai standarisasi M untuk laki-laki, dan F untuk

perempuan terhadap data jenis kelamin yang berpotensi memiliki definisi yang beragam dari sumber-sumber yang berbeda.

2. Membuat *surrogate key* dan *foreign key* yang berhubungan dengan *surrogate key* yang berkesesuaian.
3. Melakukan perhitungan atau *summarize* sebagai nilai baru untuk mengisi table fakta pada *data warehouse*.
4. Memisahkan sebuah kolom menjadi beberapa kolom terpisah (Contohnya meletakkan sebuah *comma-separated list* yang dispesifikasikan sebagai sebuah *string* dalan satu kolom yang akan diletakkan pada kolom-kolom terpisah).
5. Menggunakan berbagai bentuk validasi data baik yang sederhana maupun kompleks.

3. Load

Fase *load* merupakan tahapan yang berfungsi untuk memasukkan data ke dalam target akhir, yang dalam konteks ini adalah *data warehouse*. Data ini berasal dari proses sebelumnya yaitu proses *transformasi*, setelah data yang dihasilkan telah sesuai dengan kondisi pada *data warehouse*, proses *load* akan berjalan, data dari *staging area* akan dipindahkan ke lokasi yang bersesuaian pada *data warehouse*.

2.1.5.9 Perbedaan *Data Warehouse* dengan OLTP

Menurut Connoly dan Begg (2005, p1153), biasanya sebuah organisasi mempunyai beberapa sistem *Online Transaction Processing* (OLTP) yang berbeda untuk setiap proses bisnis, seperti pengawasan persediaan (*Inventory Control),* pesanan pelanggan (*Invoicing customer)* dan tingkat penjualan. Sistem ini menghasilkan data operasional yang detil, terbaru dan selalu berubah. Sistem OLTP optimal jika digunakan untuk sejumlah transaksi yang dapat diramalkan (*predictable),* berulang *(repetitive),* dan sering diperbaharui *(update intensive).* Data OLTP diorganisasikan berdasarkan syarat-syarat dari transaksi dihubungkan dengan aplikasi bisnis dan mendukun keputusan per hari dalam sejumlah besar operasional *user* yang konkruen.

Umumnya organisasi hanya mempunyai satu *data warehouse* yang menyimpan data secara historis, detil dan ringkasan dengan beberapa tingkatan dan sangat jarang berubah. *Data Warehouse* didesain untuk mendukung transaksi yang tidak dapat diramalkan (*unpredictable),* dan memerlukan jawaban untuk *query* khusus (*ad hoc),* tidak terstruktur dan *heuristic*. *Data warehouse* diorganisasikan berdasarkan pada syarat-syarat *query* yang potensial dan mendukung keputusan strategis jangka panjang dari sejumlah kecil *user* tingkat manajerial.

Di bawah ini adalah tabel perbandingan antara sistem OLTP dengan sistem Data Warehouse (Connoly dan Begg, 2005, p1153) :

Tabel 1 Perbandingan Sistem Data Warehouse dan Sistem OLTP

Sistem OLTP	**Sistem *Data Warehouse***
Mengandung data terkini	Mengandung data historis (data lama)
Menyimpan data yang rinci	Menyimpan data yang rinci, sedang dan ringkas
Data bersifat dinamis	Data bersifat statis
Prosesnya berulang	Prosesnya tidak terstruktur, ditujukan untuk maksud tertentu
Digunakan untuk transaksi	Digunakan untuk analisis
Transaksi tingkat tinggi	Transaksi tingkat menengah sampai rendah
Berorientasi pada aplikasi	Berorientasi pada subjek
Penggunaannya dapat diprediksi	Penggunaanya tidak dapat diprediksi sebelumnya
Mendukung keputusan harian	Mendukung keputusan yang bersifat strategis
Digunakan oleh banyak *user* operasional	Digunakan oleh sedikit *user* manajerial

2.1.6 Agregasi

Menurut Mallach (2000, p514), Agregasi merupakan kumpulan dari elemen-elemen pada beberapa dimensi dari *database*. Toko dapat diagregasikan ke dalam daerah; hari dapat diagregasikan ke dalam minggu, bulan dan kuartal; dan produk dapat diagregasikan ke dalam kategori.

2.1.7 Denormalisasi

Menurut Connoly dan Begg (2005, p520), Denormalisasi mengarah pada suatu situasi dimana 2 relasi digabungkan menjadi satu relasi baru, dan relasi baru tersebut masih ternormalisasi, namun mengandung *null* yang lebih sedikit dari relasi yang lama. Denormalisasi biasanya digunakan pada saat kinerja tidak

memuaskan dan suatu relasi yang memiliki tingkat *update* rendah serta tingkat *query* tinggi.

Dalam melakukan denormalisasi penyimpanan data dalam *database* akan melanggar ketentuan dalam normalisasi terutama *Third Normal Form* (3NF) yang bertujuan untuk menghilangkan redundansi data. Namun, jika normalisasi menghabiskan waktu dalam memberikan suatu informasi dari tabel yang diinginkan, dan akan lebih hemat jika disimpan dalam sebuah tabel.

2.1.8 Matriks

Matriks menurut Kimball (2002, p398) adalah Alat yang digunakan untuk membuat, mendokumentasikan, dan berkomunikasi dengan arsitektur data, dimana baris-baris pada matriks meng-identifikasikan proses bisnis perusahaan dan kolom-kolomnya merepresentasikan dimensi-dimensi yang sesuai dengan proses bisnis perusahaan. Perpotongan dari dimensi-dimensi yang relevan dengan masing-masing proses bisnis diberi tanda untuk menunjukkan ada hubungan antara dimensi dengan proses bisnis tertentu.

Matriks merupakan suatu perangkat yang membantu penggunanya baik dalam hal perencanaan maupun proses komunikasi. Meskipun matriks hanya terdiri dari baris dan kolom, namun mampu mendefinisikan keseluruhan arsitektur data bagi *data warehouse.* Matriks memperlihatkan keseluruhan rencana yang dirancang dalam suatu bentuk yang ringkas sehingga dapat digunakan untuk membantu senior IT dan Manajemen bisnis dalam meng-komunikasikan rancangan rencana.

Membuat Bus Matriks *Data Warehouse* adalah salah satu *up-front deliverables* yang paling penting dari implementasi *data warehouse*. Ini adalah *a*

hybrid resource yang merupakan bagian desain alat teknis, bagian alat manajemen proyek, dan bagian alat komunikasi (Kimball, 2002, p81).

2.2 Teori Khusus

2.2.1 Penjualan

Menurut Mulyadi (2001, p202), Penjualan dibedakan menjadi dua bagian yaitu penjualan tunai dan penjualan kredit. Penjualan tunai dilakukan ketika barang dan jasa diserahkan oleh perusahaan kepada pembeli ketika perusahaan sudah menerima kas dari pembeli, sedangkan dalam penjualan kredit jika order dari pelanggan telah dipenuhi dengan pengiriman barang atau jasa, untuk jangka waktu tertentu perusahaan memiliki piutang kepada pelanggannya.

Fungsi yang terkait dalam sistem penjualan, adalah sebagai berikut :

1. Fungsi Kredit

   Fungsi ini yang bertanggung jawab atas pemberian kartu kredit kepada pelanggan terpilih. Sebelum pelanggan diberikan kartu kredit, ia harus mengajukan permintaan menjadi anggota kartu kredit perusahaan dengan mengisi formulir permintaan menjadi anggota. Fungsi ini melakukan pengumpulan informasi tentang kemampuan keuangan calon anggota dengan meminta fotokopi rekening Koran bank. Keterangan gaji atau pendapatan calon anggota dari perusahaan tempat ia bekerja dan dari sumber-sumber lain.

2. Fungsi Penjualan

   Fungsi ini bertanggung jawab dalam memenuhi kebutuhan barang pelanggan. Fungsi penjualan mengisi faktur penjualan untuk memungkinkan

fungsi gudang menyediakan barang dan fungsi pengiriman menyerahkan barang kepada pelanggan.

3. Fungsi Gudang

   Fungsi ini bertanggung jawab menyediakan barang yang diperlukan oleh pelanggan sesuai dengan yang tercantum dalam tembusan faktur penjualan.

4. Fungsi Pengiriman

   Fungsi ini bertanggung jawab menyerahkan barang yang kualitas, mutu dan spesifikasinya sesuai dengan yang tercantum dalam tembusan faktur penjualan yang diterima oleh fungsi penjualan. Fungsi ini juga bertanggung jawab memperoleh tanda tangan dari pelanggan atas faktur penjualan sebagai bukti yang diterimanya barang yang dibeli oleh pelanggan.

5. Fungsi Akuntansi

   Bertanggung jawab untuk mencatat transaksi bertambahnya piutang kedalam kartu piutang berdasarkan faktur penjualan yang diterima dari fungsi pengiriman. Juga bertanggung jawab dalam pencatatan transaksi penjualan ke dalam jurnal penjualan.

6. Fungsi Penagihan

   Fungsi ini bertanggung jawab membuat surat tagihan secara periode pada pemegang kartu kredit.

2.2.2 Pembelian

Menurut Mulyadi (2001, p299), Pembelian adalah suatu usaha pengadaan barang yang diperlukan perusahaan. Transaksi dapat digolongkan menjadi dua, yaitu pembelian local dan pembelian impor. Pembelian local adalah pembelian dari pemasok dalam negeri, sedangkan impor adalah pembelian dari pemasok luar negeri. Fungsi yang terkait dengan pembelian adalah :

1. Fungsi Gudang

   Bertanggung jawab untuk mengajukan permintaan pembelian sesuai dengan posisi persediaan barang yang ada di gudang dan untuk menyimpan barang yang telah diterima oleh fungsi penerimaan.

2. Fungsi Pembelian

   Bertanggung jawab untuk memperoleh informasi mengenai harga barang. Menentukan pemasok yang dipilih dalam pengadaan barang dan mengeluarkan order pembelian kepada pemasok yang dipilih.

3. Fungsi Penerimaan

   Bertanggung jawab untuk melakukan pemeriksaan terhadap jenis, mutu dan kuantitas barang yang diterima dari pemasok guna menentukan dapat atau tidaknya barang tersebut diterima oleh perusahaan. Fungsi ini juga bertanggung jawab untuk menerima barang dari pembeli yang berasal dari transaksi retur penjualan.

4. Fungsi Akuntansi

   Fungsi yang terkait adalah fungsi pencatatan utang dan persediaan barang. Fungsi pencatatan utang berfungsi untuk mencatat transaksi ke dalam

register bukti kas keluar. Fungsi persediaan barang bertanggung jawab untuk mencatat harga produksi barang yang dibeli ke dalam kartu persediaan.

2.2.3 Persediaan

Menurut Mulyadi (2001, p553), persediaan dalam perusahaan manufaktur terdiri dari : persediaan produk jadi, persediaan produk dalam proses, persediaan bahan baku, persediaan bahan penolong, persediaan habis pakai pabrik dan persediaan suku cadang. Sedangkan dalam perusahaan dagang, persediaan hanya terdiri dari persediaan barang dagangan. Persediaan dalam perusahaan manufaktur bersangkutan dengan transaksi intern perusahaan dan transaksi yang menyangkut pihak luar perusahaan (penjualan dan pembelian), sedangkan persediaan pada perusahaan dagang hanya menyangkut intern perusahaan saja.

Sistem dan prosedur yang bersangkutan dengan sistem akuntansi persediaan adalah :

1. Prosedur pencatatan produk jadi.
2. Prosedur pencatatan harga pokok produk jadi yang dijual.
3. Prosedur pencatatan harga pokok produk jadi yang diterima kembali dari pembeli.
4. Prosedur pencatatan tambahan dan penyesuaian kembali harga pokok persediaan produk dalam proses.
5. Prosedur pencatatan harga pokok persediaan yang dibeli.
6. Prosedur pencatatan harga pokok persediaan yang dikembalikan kepada pemasok.
7. Prosedur permintaan dan pengeluaran barang gudang.

8. Prosedur pencatatan tambahan harga pokok persediaan karena pengembalian barang gudang.
9. Sistem perhitungan fisik persediaan.