BAB 2

LANDASAN TEORI

2.1 Konsep Dasar *Data warehouse*

2.1.1 Pengertian *Data warehouse*

Pengertian *data warehouse* menurut Inmon (2002, p31), "*a data warehouse is a subject oriented, nonvolatile, time variant collection of data in support of management's decisions* " atau dapat diartikan " *data warehouse* adalah koleksi data yang mempunyai sifat berorientasi subjek, terintegrasi , tidak mengalami perubahan dan mempunyai variasi waktu yang digunakan untuk mendukung proses pengambilan keputusan manajemen".

Menurut Post (2002, p548) *data warehouse* adalah spesialisasi basis data yang dioptimasi untuk memenuhi permintaan manajemen, data diekstrak dari sistem *online transaction processing* (OLTP), kemudian dibersihkan dan dioptimasisasi untuk pencarian dan analisis.

Jadi dapat disimpulkan bahwa *data warehouse* adalah kumpulan data yang telah diringkas dan terintegrasi dari data operasional maupun data *external*, yang memiliki karakteristik *subject-oriented*, *integrated*, *nonvolatile* dan *time variant* yang berguna dalam pengambilan keputusan.

2.1.2 Tujuan Perancangan *Data warehouse*

Data warehouse yang digunakan selama ini memberikan kemudahan dan keuntungan karena *data warehouse* biasanya digunakan untuk melakukan empat tugas

yang berbeda. Menurut Williams (1998, p533), keempat tugas *data warehouse* tersebut adalah sebagai berikut :

1. Pembuatan Laporan

   Pembuatan laporan merupakan salah satu kegunaan *data warehouse* yang paling umum. Dengan menggunakan *query* sederhana dalam *data warehouse*, dapat dihasilkan informasi per tahun, per semester, per bulan, dan bahkan per hari.

2. *On-Line Analytical Processing* (OLAP)

   Data warehouse digunakan dalam melakukan analisis bisnis untuk mengetahui kecenderungan pasar dan faktor-faktor penyebabnya, karena dengan adanya *data warehouse*, semua informasi baik detail maupun hasil *summary* yang dibutuhkan dalam proses analisa mudah didapat. Dalam hal ini *data warehouse* merupakan *tools* handal untuk analisa data yang kompleks.

   OLAP mendayagunakan konsep data multidimensi dan memungkinkan pemakai untuk menganalisa data sampai mendetail, tanpa mengetikkan satu pun perintah SQL. Hal ini dimungkinkan karena pada konsep data multidimensi, data berupa fakta yang sama bisa dilihat dengan menggunakan dimensi yang berbeda. Fasilitas lain yang ada pada *tools* perangkat lunak OLAP adalah *drill-down* dan *roll-up*. *Drill-down* adalah kemampuan untuk melihat detail dari suatu informasi yang ditampilkan sedangkan *roll-up* adalah kebalikan dari *drill-down*.

3. *Data mining*

   Data mining adalah proses untuk mencari informasi dan pengetahuan baru dengan cara menggali (*mining*) data yang berjumlah banyak pada *data warehouse*, dengan menggunakan kecerdasan buatan (*Artificial Intelligence*),

statistik, dan matematika. *Data mining* merupakan teknologi yang diharapkan bisa menjembatani komunikasi antara data dan pemakainya.

Beberapa solusi yang bisa diselesaikan dengan *data mining* diantaranya :

- Menembak target pasar

  Data mining dapat melakukan pengelompokan (*clustering*) dari model-model pembeli dan melakukan klasifikasi terhadap setiap pembeli sesuai dengan karakteristik yang diinginkan seperti kesukaan yang sama, tingkat penghasilan yang sama, kebiasaan membeli dan karakteristik lainnya.

- Melihat pola beli pemakai dari waktu ke waktu

  Data mining dapat digunakan untuk melihat pola beli seseorang dari waktu ke waktu.

- *Cross-market Analisis*

  Data mining dapat dimanfaatkan untuk melihat hubungan antara penjualan satu produk dengan produk lainnya.

- *Profil Customer*

  Data mining dapat membantu pengguna untuk melihat profil pembeli sehingga dapat diketahui pembeli tertentu suka membeli produk apa saja.

- Informasi *Summary*

  Data mining dapat dimanfaatkan untuk membuat laporan summary yang bersifat multidimensi yang dilengkapi dengan informasi statistik lainnya.

4. Proses Informasi Eksekutif

Data warehouse digunakan untuk mencapai 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. Hal ini akan mempermudah proses pengambilan keputusan. Informasi dan data pada laporan *data warehouse* menjadi target informatif bagi *user*, dimana *user* disini adalah pihak eksekutif.

2.1.3 Karakteristik *Data warehouse*

Menurut Inmon (2002, p31) dapat diketahui bahwa sebuah *data warehouse* mempunyai karakteristik, antara lain :

1. *Subject Oriented* (Berorientasi Subyek)
2. *Integrated* (Terintegrasi)
3. *Non – Volatile* (Tidak Dapat Berubah)
4. *Time – Variant* (Variasi Waktu)

2.1.3.1 *Subject Oriented*

Data warehouse bersifat *subject oriented* artinya sebuah *data warehouse* dirancang dan dibangun untuk memenuhi kebutuhan analisis data berdasarkan subjek tertentu. Misalnya analisis mengenai data penjualan dalam sebuah organisasi bisnis.

2.1.3.2 *Integrated*

Data warehouse bersifat *integrated* artinya *data warehouse* harus menyimpan data yang berasal dari sumber-sumber yang terpisah ke dalam suatu format yang konsisten dan saling terintegrasi satu dengan yang lainnya. Dengan demikian data tidak bisa dipecah-pecah karena data yang ada merupakan satu kesatuan yang menunjang keseluruhan konsep *data warehouse* itu sendiri.

Data warehouse harus dapat memecahkan masalah-masalah seperti konflik penamaan variabel dan inkonsistensi diantara ukuran-ukuran yang dapat dipakai didalamnnya dengan cara konsistensi dalam pemberian nama, penentuan pengukuran ukuran dari tipe variabel, struktur coding, serta penentuan atribut data secara spesifik.

2.1.3.3 *Non-volatile*

Non-volatile memiliki arti ketika data sudah disimpan ke dalam sebuah *data warehouse,* data harus tidak boleh dirubah atau tidak boleh ada perubahan didalamnya. Hal ini sangat logis karena tujuan dari *data warehouse* ini adalah memungkinkan untuk menganalisa data histori tentang apa yang terjadi.

2.1.3.4 *Time Variant*

Data warehouse menyimpan sejarah (*historical data*). Waktu merupakan tipe atau bagian data sangat penting didalam *data warehouse*. Di dalam *data warehouse* sering disimpan macam-macam waktu, seperti waktu suatu transaksi terjadi, transaksi dirubah, dan transaksi dibatalkan. Data yang disimpan juga hampir selalu disimpan dalam berbagai versi, misalnya terjadi perubahan definisi kode pos, maka yang lama dan yang baru ada semua didalam *data warehouse*. Jadi, *data warehouse* yang bagus adalah

yang menyimpan sejarah. Contoh *data warehouse* menyimpan semua data perusahaan dari setiap tahun sejak pertama perusahaan tersebut berdiri.

Terlihat, bahwa keempat karakteristik ini saling terkait, semuanya harus diimplementasikan agar suatu *data warehouse* dapat berjalan efektif untuk mendukung pengambilan keputusan. Dan, implementasi keempat karakteristik ini membutuhkan struktur data dari *data warehouse* yang berbeda dengan *database* sistem operasional.

2.1.4 Anatomi *Data warehouse*

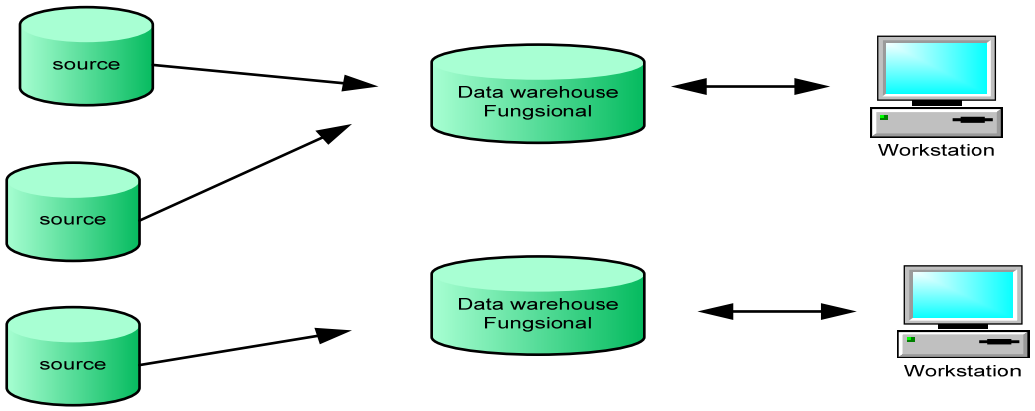
Terdapat tiga jenis dasar sistem anatomi *data warehouse*, yaitu *data warehouse* fungsional, *data warehouse* terpusat, dan *data warehouse* terdistribusi.

1. *Data warehouse* Fungsional

Menurut http://www.etfinancial.com/dataglossary.htm *fungsional data warehouse* adalah sebuah gudang (*warehouse*) yang menggambarkan data dari sistem operasional. Setiap *fungsional warehouse* menyediakan grup yang terpisah dan berbeda–beda (seperti divisi), *area fungsional* (seperti *manufacturing*), unit geografi atau grup produk *marketing*.

Kelebihan dari bentuk fungsional ini adalah sistem mudah dibangun dengan biaya yang relatif murah dan dapat memberikan kemampuan sistem pengumpulan data yang terbatas pada kelompok *user*.

Sedangkan kekurangan dari bentuk fungsional ini adalah resiko kehilangan konsistensi dari data dan juga terbatasnya kemampuan dalam pengumpulan data bagi *user*.



Gambar 2.1 *Data warehouse* Fungsional

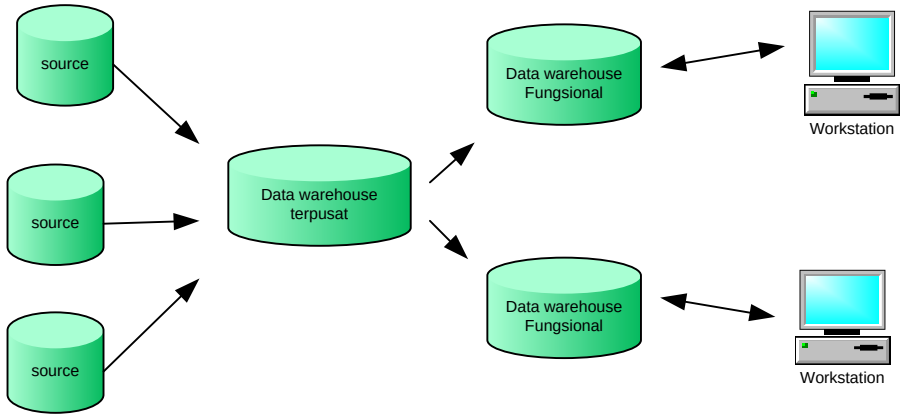
(http://myhut.org/public/datawarehouse.doc)

2. *Data warehouse* Terpusat

Menurut http://www.geekinterview.com/kb/Central-Data-Warehouse.html *Central data warehouse* adalah suatu database fisik yang mana mengandung data usaha khusus fungsi daerah, instansi, cabang, divisi atau seluruh perusahaan. Dan disimpan dalam suatu tempat dan dibuat berdasarkan ekstrak data oprasional serta terintegrasi.

Kelebihan *data warehouse* terpusat adalah data benar-benar terintegrasi. Sistem ini mengharuskan data dikirim tepat pada waktunya. Disamping itu, *user* hanya dapat mengambil data dari pusat pengumpulan saja dan tidak dapat berhubungan secara langsung dengan pemasok datanya sendiri

Kekurangan *data warehouse* terpusat adalah pada penerapannya membutuhkan biaya pemeliharaan yang tinggi atas sistem pengumpulan data yang besar. Selain itu diperlukan waktu yang lama untuk membangun sistem tersebut.



Gambar 2.2 *Data warehouse* Terpusat

(http://myhut.org/public/datawarehouse.doc)

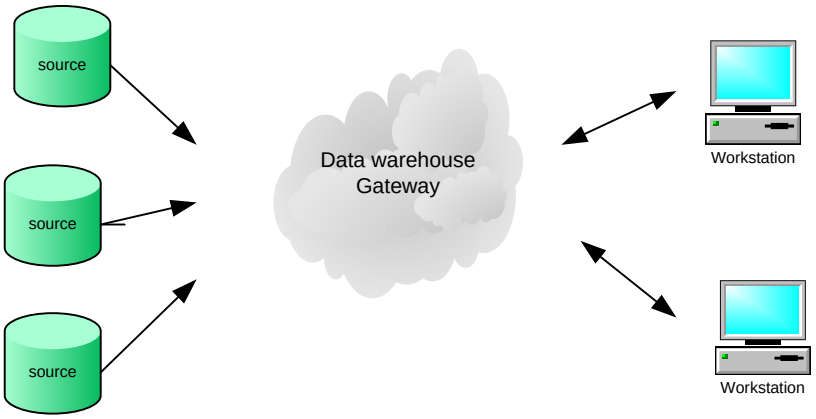
3. *Data warehouse* Terdistribusi

Menurut http://www.ittelkom.ac.id/library/index.php?view=article&catid=20%3Ainformatika&id=484%3Adata-warehouse-&option=com_content&Itemid=15

Distributed data warehouse adalah kumpulan *data store* yang dibangun secara terpisah yang digabungkan secara fisik melalui jaringan. Tujuannya adalah agar komponen-komponen yang terpisah ini terlihat sebagai satu kesatuan utuh sebuah sistem data warehouse . Suatu *enterprise data warehouse* dapat dibentuk dari kumpulan *data mart* yang terpisah, jadi tidak selalu membentuk sistem yang terpusat tetapi juga bisa terdistribusi. Dengan kecenderungan *data-oriented*, data pada suatu perusahaan atau organisasi seharusnya merupakan data yang *widely-shareable*.

Kelebihan *data warehouse* terdistribusi adalah kelebihannya dalam mengakses data dari luar perusahaan yang lebih mengalami sinkronisasi terlebih dahulu dan tetap menjaga konsistensinya. Hal ini dikarenakan *data warehouse* menggunakan teknologi *client – server* untuk mengambil data dari berbagai sumber.

Kekurangan *data warehouse* terdistribusi adalah pada penerapannya membutuhkan biaya yang sangat besar dan juga kompleks untuk diterapkan karena sistem operasinya digunakan secara terpisah.



Gambar 2.3 *Data warehouse* Terdistribusi

(http://myhut.org/public/datawarehouse.doc)

2.1.5 Struktur *Data warehouse*

Sebuah *data warehouse* memiliki beberapa struktur, seperti :

1. *Physical Data warehouse*

   Tempat dimana semua data untuk *data warehouse* disimpan bersama metadata dan proses logis untuk *scrubbing* (menghapus), *organizing* (mengatur), *packaging* (mengumpulkan) dan proses dari detail data.

2. *Logical Data warehouse*

   Berisikan metadata termasuk *enterprise rules* dan proses logis untuk*scrubbing* (menghapus), *organizing* (mengatur), *packaging* (mengumpulkan) dan proses data. Tetapi tidak berisikan data yang aktual. Disamping itu juga berisikan informasi

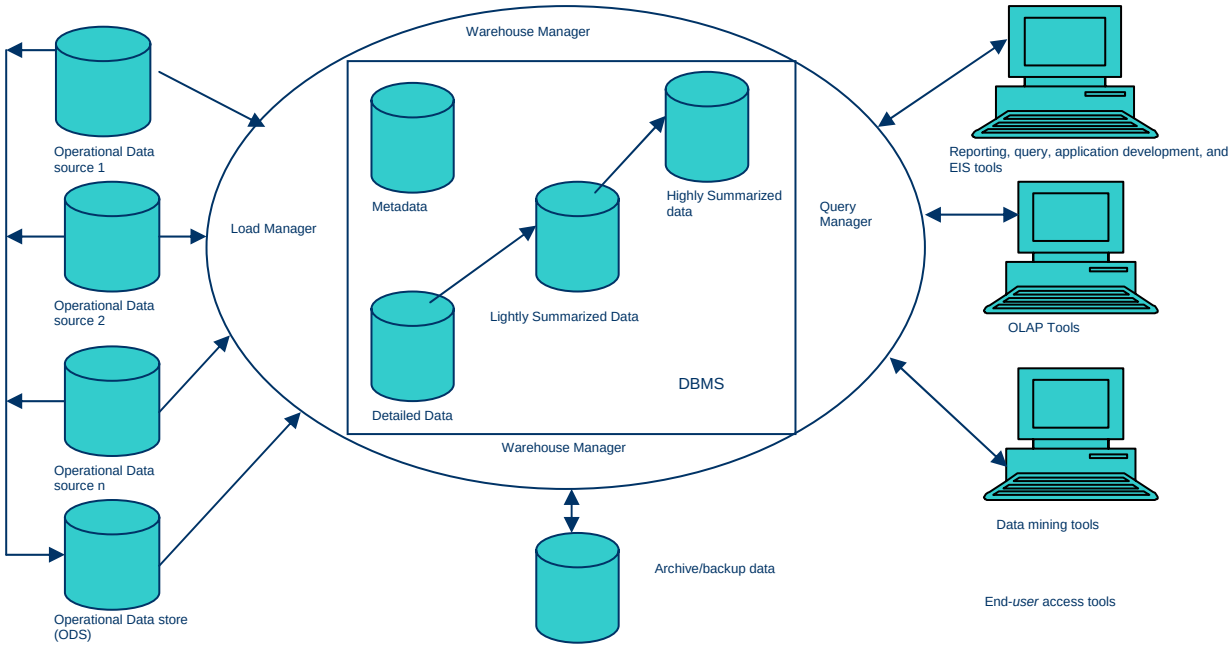
yang diperlukan untuk mengakses data dimana saja.

3. *Data Mart*

   Adalah suatu bagian dari *data warehouse* yang dapat mendukung pembuatan laporan dan analisa data pada suatu unit, bagian atau operasi pada perusahaan. Sebagai bagian dari proses pengembangan *data warehouse* yang selalu berulang, sebuah perusahaan perlu membangun sebuah rangkaian *physical data marts* dan menghubungkannya melalui *enterprise-wide logical data warehouse* atau dimasukkan dari *single physical data warehouse.*

   (http://www.ies.aust.com/papers/dw.htm)

2.1.6 Arsitektur *Data warehouse*



Gambar 2.4 Arsitektur *Data warehouse*

(http://myhut.org/public/datawarehouse.doc)

Menurut Connoly dan Begg (2002, p1052), komponen utama *data warehouse*, antara lain :

1. Data Operasional

   Data operasional adalah data yang digunakan untuk mendukung proses bisnis sehari-hari.

2. *Operational Data Store* (ODS)

   Operatinal data store adalah tempat penyimpanan data operasional yang bersifat *current* dan terintegrasi yang digunakan untuk analisis. Atau dengan kata lain, ODS mendukung proses transaksi operasional maupun proses analisis. Dengan adanya ODS maka pembangunan *data warehouse* menjadi lebih mudah karena ODS dapat menyediakan data yang telah diekstrak dari sumber dan telah dibersihkan sehingga proses pengintegrasian dan restrukturisasi data untuk *data warehouse* menjadi lebih sederhana.

3. *Load Manager*

   Disebut juga komponen *front end* menangani semua operasi yang berhubungan dengan fungsi *extract data* (mengambil data) dan fungsi *loading data* (menaruh data) ke dalam *data warehouse*.

4. *Warehouse Manager*

   Warehouse manager menangani semua operasi yang berhubungan dengan *management* data dalam *data warehouse*. Operasi-operasi yang dijalankan oleh *warehouse manager* mencakup :

   a. Analisis data untuk menjaga konsistensi data.
   b. Melakukan transformasi dan penggabungan sumber data dari tempat penyimpanan sementara ke dalam tabel-tabel *data warehouse.*

c. Melakukan denormalisasi.

d. Melakukan agregasi.

e. Menyimpan (*archive*) dan *back-up* data.

5. *Query Manager*

Query manager (disebut juga komponen *backend*) menangani semua operasi yang berhubungan dengan management permintaan *user* (*user queries*). Operasi yang dijalankan oleh *query manager* meliputi kegiatan mengarahkan permintaan ke tabel-tabel data yang tepat dan melakukan penjadwalan eksekusi terhadap permintaan.

6. *Detailed Data*

Dalam *data warehouse*, area ini adalah tempat penyimpanan semua *detailed data* dalam skema basis data. *Detailed data* dibagi menjadi 2, yaitu *current detail data* (tempat penyimpanan semua *detailed data* yang bersifat *current*) dan *old detailed data* (tempat penyimpanan semua *detailed data* yang bersifat *old*).

7. *Lightly and Highly Summerized Data*

Area ini adalah tempat penyimpanan sementara data predefinisi yang teringkas secara *light d*an *high* (*predefined lightly and highly summarized*) yang dihasilkan oleh *warehouse manager*. Tujuan dari ringkasan informasi ini adalah untuk mempercepat tanggapan terhadap permintaan *user*. Ringkasan data di-*update* secara berkala seiring dengan bertambahnya data dalam *data warehouse*.

8. *Archive* / *Backup* data

Dalam *data warehouse*, area ini digunakan untuk menyimpan *detailed* data dan data yang telah diringkas. Tujuannya adalah untuk penyimpanan (*archiving*) dan *backup*.

Data kemudian ditransfer ke media penyimpanan seperti *magnetic tape* atau *optical disk.*

9. Metadata

   Digunakan untuk menyimpan semua definisi metadata (keterangan tentang data) yang digunakan dalam seluruh proses *warehouse*.

   Metadata digunakan untuk berbagai tujuan, antara lain :

   a. proses *extracting* dan *loading*
   b. proses *warehouse management*
   c. sebagian *proses query management*

10. End-User Access Toolss

   End-user access toolss adalah *tools* yang memanfaatkan kegunaan dari *data warehouse*. Kegunaan *data warehouse* tersebut, antara lain untuk pembuatan laporan, OLAP, *data mining* dan proses informasi eksekutif.

2.1.7 Infrastruktur *Data warehouse*

Infrastruktur *data warehouse* terdiri dari *software, hardware*, pelatihan-pelatihan dan komponen-komponen lainnya yang memberikan dukungan yang dibutuhkan untuk mengimplementasikan arsitektur *data warehouse* Poe (1998, p43). Salah satu instrumen yang mempengaruhi keberhasilan pengembangan *data warehouse* adalah pengidentifikasian arsitektur mana yang terbaik dan infrastruktur yang dibutuhkan.

Arsitektur dan infrastruktur sangat erat hubungannya. Arsitektur yang sama mungkin akan membutuhkan infrasturktur yang berbeda, tergantung pada lingkungan perusahaan ataupun organisasi.

2.1.8 Perancangan *Data warehouse*

2.1.8.1 Model Dimensional

Merupakan sebuah perancangan logical yang bertujuan untuk menampilkan data dalam bentuk standar dan intuitif yang memperbolehkan akses dengan performa yang tinggi.

Model dimensional mengunakan konsep model hubungan antar *entity* (ER) dengan beberapa batasan yang penting. Setiap model dimensi terdiri dari sebuah tabel dan sebuah komposit *primary key* disebut dengan tabel fakta dan satu set tabel yang lebih kecil disebut tabel dimensi. Setiap tabel dimensi memiliki sebuah *simple primary* key yang merespon tepat pada satu komponen *composite key* di tabel fakta. Dengan kata lain, *primary key* dari sebuah tabel fakta terdiri dari atas dua *foreign key*. Struktur karakteristik ini disebut skema binatang atau *join* bintang.

Model dimensional yang sering digunakan adalah skema bintang atau *snowflake* yang mudah dimengerti dan sesuai dengan kebutuhan bisnis.

2.1.8.2 Skema Bintang

Skema bintang merupakan perancangan yang memiliki struktur sederhana dengan tabel-tabel yang relatif dan penggabungan yang telah diketahui Poe (1998, p191). Skema bintang merupakan suatu rancangan *database* pada *data warehouse* yang menggambarkan hubungan yang jelas antara struktur tabel fakta dan tabel dimensi. Skema ini dapat dibaca dengan mudah oleh analis maupun pemakai yang tidak biasa dengan struktur *database*.

Skema bintang memiliki beberapa keuntungan yang tidak terdapat pada struktur relasional biasa. Berikut merupakan keuntungan dari penggunaan skema bintang :

1. Respon data lebih cepat daripada perancangan *database* operasional.
2. Mempermudah dalam hal modifikasi atau pengembangan *data warehouse* yang terus-menerus.
3. *End-user* dapat menyesuaikan cara berpikir dan menggunakan data.
4. Menyederhanakan pemahaman dan penelusuran metadata bagi pemakai dan pengembang.

2.1.8.3 Perancangan Skema Bintang

Skema bintang terdiri dari dua macam tabel, yaitu tabel fakta (*fact table*) dan tabel dimensi (*dimension table*).

- Tabel Fakta

  Sering disebut *major table*, yang merupakan inti dari skema bintang dan berisi data aktual yang akan dianalisis. *Field-field* dalam fakta disebut *measure* dan biasanya berupa numerik. Selalu berisi *foreign key* dari masing-masing tabel dimensi.

- Tabel Dimensi

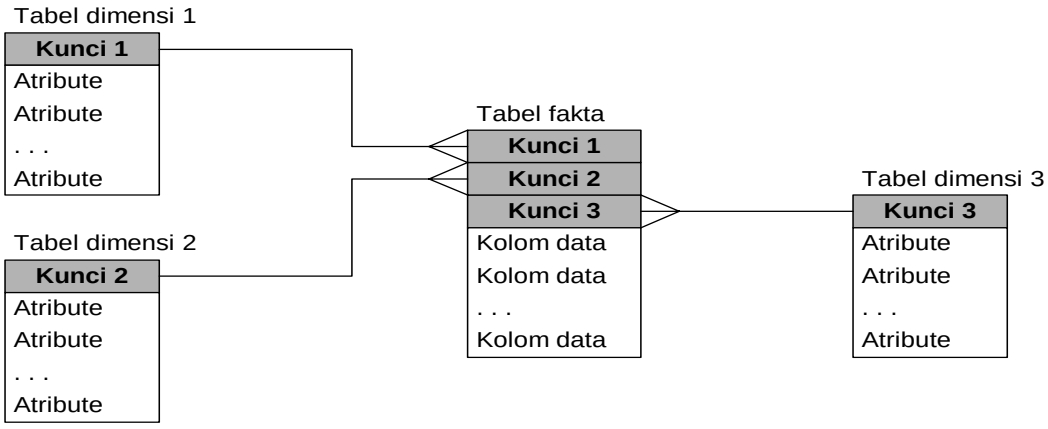
  Sering disebut *minor table*, yang merupakan tabel dari skema bintang yang menyediakan jenis perspektif dari cara pandang terhadap data. Tabel dimensi mempunyai *field-field* dari level hierarki tabel dimensi. Nama dari *field* tersebut biasanya menggunakan nama dari level dalam suatu hierarki.

2.1.8.4 Jenis-jenis Skema Bintang

Dalam penggunaannya, terdapat dua jenis skema bintang berdasarkan pada kebutuhannya, yaitu :

1. Skema bintang sederhana

   Pada skema bintang sederhana, setiap tabel harus mempunyai *primary key* yang terdiri dari satu kolom atau lebih. *Primary key* tersebut membuat masing-masing baris menjadi unik. *Primary key* pada tabel fakta terdiri dari satu atau lebih *foreign key*. *Foreign key* adalah kolom pada suatu tabel yang dinilainya didefinisikan oleh *primary key* pada tabel yang lain.



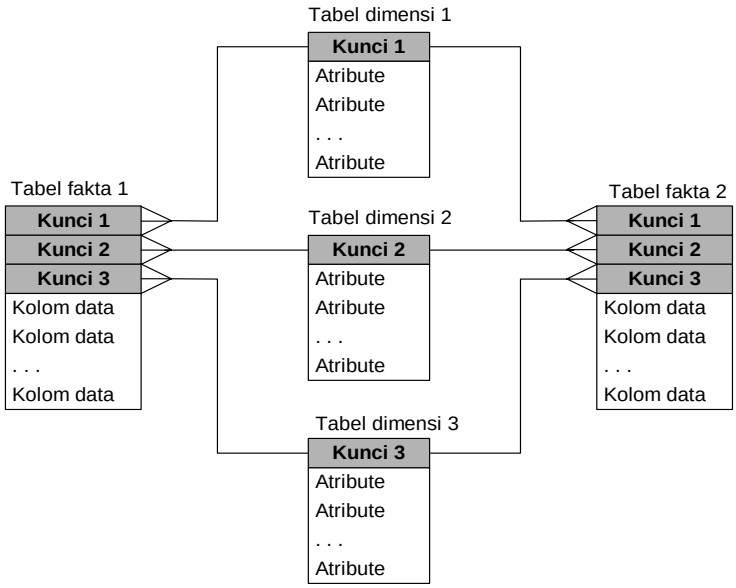
Gambar 2.5 Skema Bintang Sederhana

(http://myhut.org/public/datawarehouse.doc)

Gambar diatas menunjukkan hubungan antara satu (1) tabel fakta dan tiga (3) tabel dimensi. Tabel fakta memiliki *primarykey* yang terdiri dari tiga (3) *foreign key*, yaitu kunci ke-1, kunci ke-2, kunci ke-3, yang masing-masing merupakan *primary key* pada tabel dimensi.

2. Skema bintang dengan beberapa tabel fakta.

   Skema bintang juga dapat terdiri dari beberapa tabel fakta. Hal ini terjadi karena masing-masing tabel berisi kenyataan yang tidak saling berhubungan atau karena perbedaan waktu pemuatan data. Selain itu, digunakan untuk menampung berbagai tingkat data yang bermacam-macam, terutama jika data tersebut dalam jumlah besar.

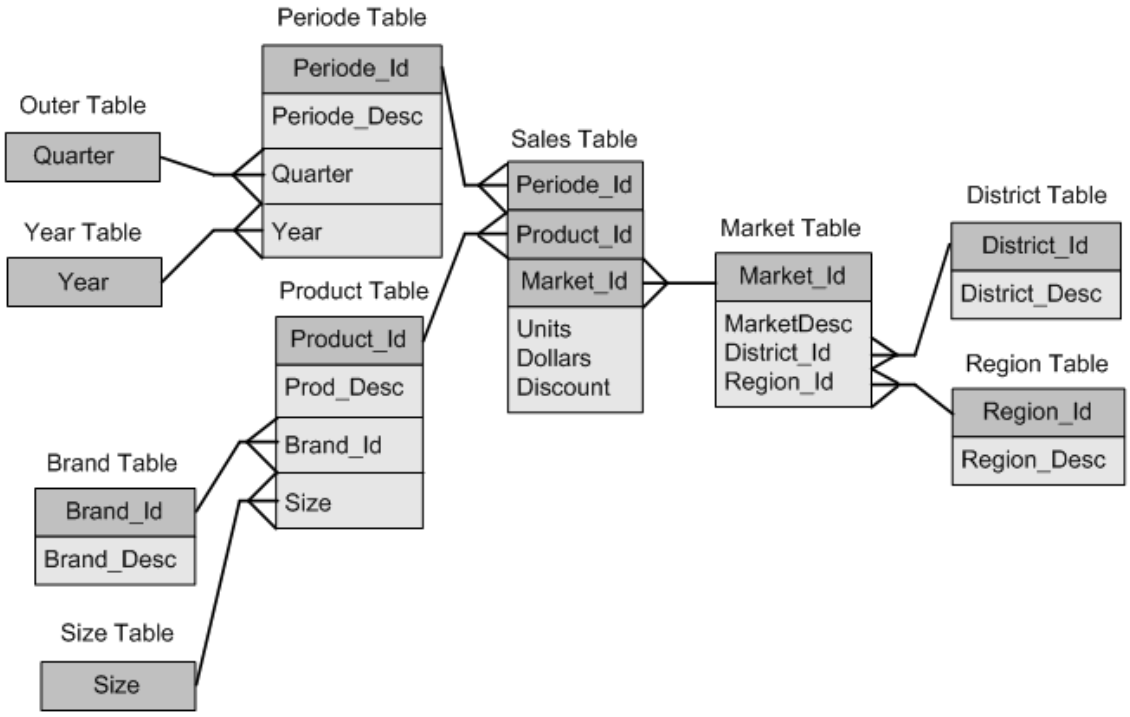


Gambar 2.6 Skema Bintang dengan banyak Tabel Fakta

(http://myhut.org/public/datawarehouse.doc)

2.1.8.5 Skema *Snowflake*

Snowflake merupakan variasi lain dari skema bintang dimana tabel dimensi dari skema bintang diorganisasikan menjadi suatu hierarki dengan melakukan normalisasi. Penggunaan tabel dimensi sangatlah menonjol, karena itulah perbedaan dasar dari skema bintang dan skema *snowflake*.



Gambar 2.7 Skema Snowflake

(Poe,1998, p199)

Ciri-ciri skema *snowflake* adalah sebagai berikut :

1. Tidak terdapat level pada tabel dimensi.
2. Tabel dimensi dinormalisasi dengan dekomposisi pada tabel atribut.
3. Setiap dimensi mempunyai satu *key* untuk setiap level pada hierarki dimensi.
4. Kunci level terendah menghubungkan tabel dimensi dengan tabel fakta dan tabel atribut level terendah.

Sedangkan beberapa keuntungan dari skema *snowflake* dapat dilihat pada pernyataan berikut :

1. Kecepatan dalam pemindahan data dari data OLAP ke dalam metadata.
2. Sebagai alat pengambil keputusan tingkat tinggi dimana dengan menggunakan skema *snowflake* seperti ini seluruh struktur dapat digunakan seluruhnya.

Banyak anggapan yang menyatakan bahwa lebih nyaman merancang dalam bentuk normal ketiga.

2.1.8.6 Agregasi

Menurut Post (2002, p544) agregasi adalah istilah umum untuk beberapa fungsi SQL yang beroperasi melewati baris-baris yang dipilih. Contoh dari agregasi antara lain : SUM, COUNT, MAX, MIN, dan AVERAGE.

Menurut Poe (1998, p136), faktor pendorong pemuatan agregasi adalah :

1. Meningkatkan kinerja (*performance*) dalam pencarian.
2. Mengurangi jumlah penggunaan kode produk universal.

Suatu agregasi yang baik dapat dibuat untuk digunakan oleh 300 *user* dalam satu hari, karena akan lebih bermanfaat jika dibandingkan dengan membuat agregasi yang membutuhkan waktu dua (2) jam tetapi hanya digunakan sekali dalam setahun olah satu (1) *user* saja.

Salah satu teknik yang harus diperhatikan adalah saat pembuatan *data warehouse*, kita tetap membutuhkan teknik *database* klasik seperti partisi tabel secara fisik. Hal ini menjadi penting bilamana *data warehouse* mencapai *gigabyte* data.

2.1.8.7 Denormalisasi

Menurut Connoly dan Begg (2002, p507), denormalisasi adalah suatu proses yang mengubah bentuk normalisasi dari *database* dengan cara penggabungan tabel dan merupakan sebuah proses yang secara sengaja dilakukan dengan melanggar peraturan bentuk normal normalisasi dengan tujuan untuk meningkatkan kinerja (*performance*) pengaksesan data yang ada.

Beberapa keuntungan dalam melakukan proses denormalisasi adalah :

1. Mengurangi jumlah relasi yang terjadi antar tabel-tabel yang harus mengalami proses pada waktu pencarian sehingga akan meningkatkan kecepatan proses *query* data.
2. Membuat struktur fisik *database* agar mudah dimengerti menurut model dimensi dari *user*. Struktur tabel yang dibuat sesuai keinginan pemakai memungkinkan terjadinya akses langsung yang sekali lagi akan meningkatkan kinerja (*performance*).

Sedangkan kelemahan dalam melakukan proses denormalisasi dapat dilihat pada berikut ini :

1. Proses denormalisasi secara tidak langsung akan membuat redudansi data.
2. Proses denormalisasi memerlukan alokasi *memory* dan *storage* (tempat penyimpanan) yang besar.

2.1.8.8 Metodologi Perancangan *Data warehouse*

Berdasarkan kutipan Connoly dan Begg (2002, p1083) metodologi yang dikemukan oleh Kimball dalam membangun *data warehouse* ada 9 tahapan, dikenal dengan *Nine-step Methodology*.

1. Memilih proses (*Choosing process*)

   Pilihlah subjek dari permasalahan yang sedang dihadapi, kemudian identifikasi proses bisnisnya. *Data mart* adalah bagian dari *data warehouse* yang pembuatan laporan dan analisis data pada suatu unit, bagian atau operasi pada perusahaan.

2. Memilih grain (*Choosing the grain*)

   Tentukan tabel fakta dan idenfikasi dimensi. Tabel fakta merupakan tabel yang mengandung angka dan *data history* dimana *key* yang dihasilkan sangat banyak karena merupakan kumpulan – kumpulan *foreign key* dan *primary key* yang ada pada masing – masing tabel dimensi yang berhubungan. Sedangan tabel dimensi adalah tabel yang berisi kategori dengan ringkasan data detail yang dapat dilaporkan, seperti laporan keuntungan pada tabel fakta, sebagai dimensi waktu (perbulan, persemester, pertahun).

3. Identifikasi dan penyesuaian dimensi (*Identifying and conforming the dimensions*)

   Identifikasi dimensi dalam detail yang secukupnya untuk mendeskripsikan sesuatu. Ketika tabel dimensi ada pada dua atau lebih *data mart*, maka tabel dimensi tersebut harus mempunyai dimensi yang sama atau salah satu merupakan *subset* dari yang lainnya. Apabila suatu tabel dimensi digunakan lebih dari satu *data mart*, maka dimensinya harus disesuaikan.

4. Memilih fakta (*Choosing the facts*)

   Tentukan fakta–fakta dari tabel fakta yang akan digunakan pada *data mart*. Fakta – fakta tersebut harus numerik dan dapat ditambah.

5. Menyimpan pre-kalkulasi pada tabel fakta (*Storing pre-calculations in the fact table*)

Setelah fakta–fakta dipilih maka lakukan pengkajian ulang untuk menentukan apakah ada fakta–fakta yang dapat diterapkan pre-kalkulasi (kalkulasi awal) dan lakukan penyimpanan pada tabel fakta.

6. Melengkapi tabel dimensi (*Rounding out the dimension tables)*

   Dalam langkah ini, kita kembali pada *dimension table* dan menambahkan gambaran teks terhadap dimensi yang memungkinkan. Gambaran teks harus mudah digunakan dan dimengerti oleh *user*. Kegunaan suatu data mart ditentukan oleh lingkup dan atribut tabel dimensi.

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

   Tentukan waktu dari pembatasan data yang diambil dan dipindahkan ke dalam tabel fakta. Seperti data perusahaan tiga tahun lalu atau lebih diambil dan dimasukkan dalam tabel fakta.

8. Melacak perubahan dari dimensi secara perlahan (*Tracking slowly changing dimensions*)

   Amati perubahan dari dimensi pada *dimension table*. Ada tiga tipe dasar dari perubahan dimensi yang perlahan, yaitu :

   a. Perubahan atribut dimensi ditulis ulang (*over write*).
   b. Perubahan atribut dimensi mengakibatkan pembuatan suatu dimensi baru.
   c. Perubahan atribut dimensi mengakibatkan sebuah atribut alternatif dibuat, jadi antar atribut yang lama dan yang baru diakses secara bersama – sama.

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

Pertimbangkan pengaruh dari perancangan fisikal, seperti keberadaan dari ringkasan (*summaries*) dan penjumlahan (*agregate*). Selain itu, masalah administrasi, *backup data*, *recovery data*, kinerja indeks dan keamanan juga merupakan faktor yang harus diperhatikan.

2.2 Aplikasi Transaksional (OLTP)

OLTP dirancang untuk memungkinkan terjadinya pengaksesan secara bersamaan oleh beberapa *user* terhadap sumber data yang sama dan mengatur proses yang diperlukan Vieira (1999, p680). OLTP mengizinkan untuk langsung mengakses ke *database*. Transaksi yang dilakukan antara lain *insert, update,* dan *delete. Database* OLTP biasanya bersifat relasional dan dalam bentuk formal yang ketiga. Dan yang terpenting, *database* OLTP dibangun untuk mampu menangani banyak transaksi dengan performa tinggi. Vieira (1999, p682).

2.3 Definisi Pembelian, Produksi, dan Persediaan

2.3.1 Konsep Pembelian

Menurut Haidar (2001,p3552), pembelian adalah mendapatkan sesuatu (barang atau jasa) dengan menukar uang yang senilai.

Sedangkan menurut Barata (2001,p29), pembelian adalah cara memperoleh sesuatu (barang atau jasa) dengan memberikan balas jasa berupa sejumlah uang yang nilainya sama dengan harga barang/jasa yang diperolehnya.

Jadi, konsep pembelian adalah suatu cara untuk mendapatkan barang atau jasa dengan memberikan balas jasa yang senilai dari barang atau jasa yang diterima.

2.3.2 Konsep Produksi

Menurut Sukanto dan Indriyo (1999, p1), produksi adalah merupakan penciptaan atau penambahan faedah bentuk, waktu dan tempat atas faktor – faktor produksi sehingga lebih bermanfaat bagi pemenuhan kebutuhan manusia.

Sedangkan menurut Render dan Haizer (2001, p2), definisi produksi adalah penciptaan dari barang dan jasa.

Jadi, konsep produksi adalah proses perubahan dari bahan baku atau barang setengah jadi menjadi barang jadi yang mempunyai nilai jual.

2.3.3 Konsep Persediaan

Menurut Sipper *et al* (1997, p206), definisi persediaan adalah *"A quality of commodity in the control of an enterprise, held for some time to satisfy some future demand"*. Yang dapat diartikan sebagai : Sejumlah komoditas yang berada dalam kendali suatu perusahaan, disimpan untuk beberapa waktu untuk memenuhi permintaan dimasa yang akan datang.

Sedangkan menurut Mulyadi (2001, p553), sistem akuntansi persediaan bertujuan untuk mencatat mutasi tiap jenis persediaan yang disimpan di gudang. Dalam perusahaan manufaktur, persediaan terdiri dari persediaan produk jadi, persediaan produk dalam proses, persediaan bahan baku, persediaan bahan penolong, persediaan bahan habis pakai pabrik, dan persediaan suku cadang.

Jadi, konsep persediaan adalah sejumlah barang milik perusahaan yang disimpan dengan maksud untuk memenuhi kebutuhan proses produksi.

2.4 Teori-Teori Penunjang

2.4.1 *Entity Relationship Diagram*

Menurut Obrien (2003, p392) ERD adalah suatu gambar yang menunjukkan suatu entitas dan hubungan antar entitasnya. Abstrak yang dipakai untuk menguraikan data adalah sebagai berikut :

- Objek *(entity)*

  Objek adalah segala sesuatu yang dapat dijelaskan dengan data, kelompok benda atau objek, umumnya diberi nama dengan kata benda. Pada diagram aliran, yang dimaksud dengan *entity* adalah *file* yang digunakan dalam sistem.

- Interaksi *(Relationship)*

  Asosiasi antara satu atau beberapa *entity*, umumnya diberi nama dengan kata kerja dasar.

- Pemilik *(Property)* atau Atribut *(Atribute)*

  Property atau *atribute* merupakan karakteristik dari suatu *entity*.

2.4.2 Bill of Material (BoM)

Menurut http://en.wikipedia.org/wiki/Bill_of_materials *bill of material is a list of the raw materials, sub-assemblies, intermediate assemblies, sub-components, components, parts and the quantities of each needed to manufacture an end item (final product).* Yang dapat diartikan: adalah daftar bahan baku, sub-pemasangan, pemasangan menengah, sub-komponen, komponen, bagian dan kuantitas masing-masing diperlukan untuk memproduksi barang akhir (produk akhir).

Sedangkan menurut http://searchmanufacturingerp.techtarget.com/sDefinition/0,,sid193_gci1353395,00.html *is a list of the parts or components that are required to build a product. The BoM provides the manufacturer's part number (MPN) and the quantity needed for each component*. Yang dapat diartikan: adalah daftar bagian-bagian atau komponen yang diperlukan untuk membangun suatu produk. Terlahir menyediakan nomor bagian produsen (MPN) dan kuantitas yang dibutuhkan untuk setiap komponen.

2.4.3 Konsep Staging Area

Staging area merupakan tempat untuk membersihkan (filter), mengubah, mengkombinasikan, dan menyediakan sumber data. Ada tiga fungsi penting yang ada dalam area ini :

a. Ekstraksi Data

Sumber data mungkin berasal dari mesin yang berbeda-beda dengan format data yang berbeda pula. Atau mungkin akan digabungkan pula dengan data dari *spreadsheet* dan data per departemen. Ekstraksi data akan menjadi sangat kompleks. Oleh karena itu, sumber data diekstrak ke dalam lingkungan fisik yang berbeda yang kemudian dimasukkan ke dalam *Staging Area.* Setelah itu data dari *Staging Area* akan diexkstraksi lagi ke dalam *Data Warehouse.*

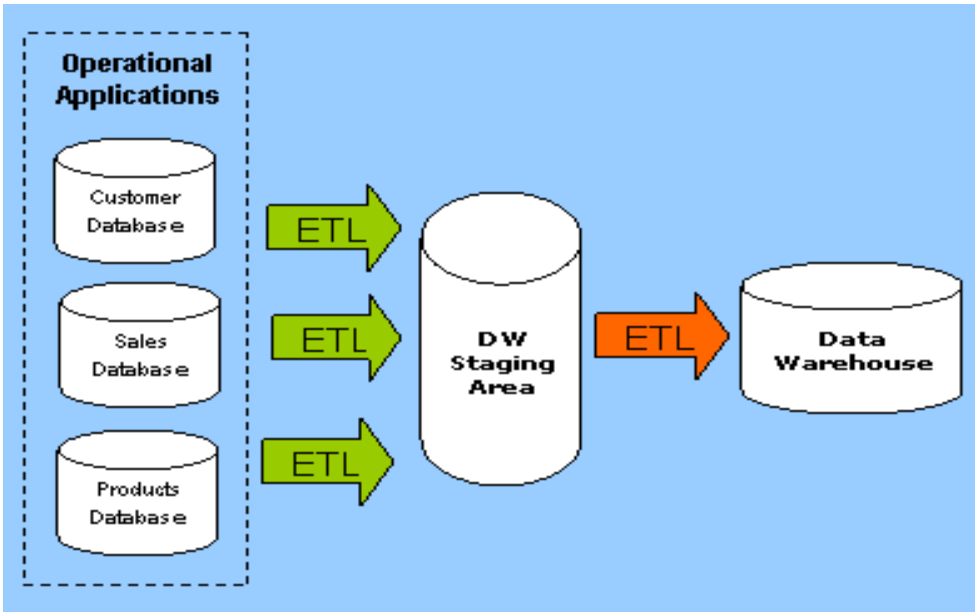
b. Transformasi Data

Hal-hal yang perlu dilakukan dalam transformasi data adalah membersihkan data yang diekstrak dari masing-masing sumber. Maksud dari pembersihan di sini adalah membenarkan kesalahan pengejaan kata, adanya perbedaan ukuran dalam sumber data, atau menyediakan nilai *default* untuk data yang hilang, atau mengeliminasi data yang

duplikat. Standarisasi tipe data dan ukuran *field* untuk setiap elemen data yang diambil dari banyak sumber adalah bagian yang paling besar dalam transformasi data.

c. Loading Data

Setelah menyelesaikan perancangan dan konstruksi *data warehouse,* dilakukan *loading* data ke dalam *data warehouse.*



Gambar 2.8 *Staging Area*

(http:/data-warehouses.net/architecture/staging.html)

Data diekstrak dari system sumber, kemudian data tersebut ditransform lalu diloading ke dalam *staging area*. Begitu di *staging area*, data akan dibersihkan, distandarkan kemudian diformat ulang untuk siap diloading ke dalam *data warehouse*.

Dengan munculnya *data warehouse*, konsep transformasi telah menyediakan tingkat tinggi kualitas dan keseragaman data. Konvensional (pra-*data warehouse*) *Staging area* digunakan untuk menjadi dataran timbunan dari data produksi. Sebuah *Staging area* dengan ekstraksi dan transformasi adalah yang terbaik untuk menghasilkan

tingkat kualitas informasi transaksi, hal ini disebabkan karena lalu lintas pengambilan *database* tidak berat, dan penyamaan format dari cabang, sebelum nantinya data–data tersebut dimasukkan ke dalam *data warehouse*.

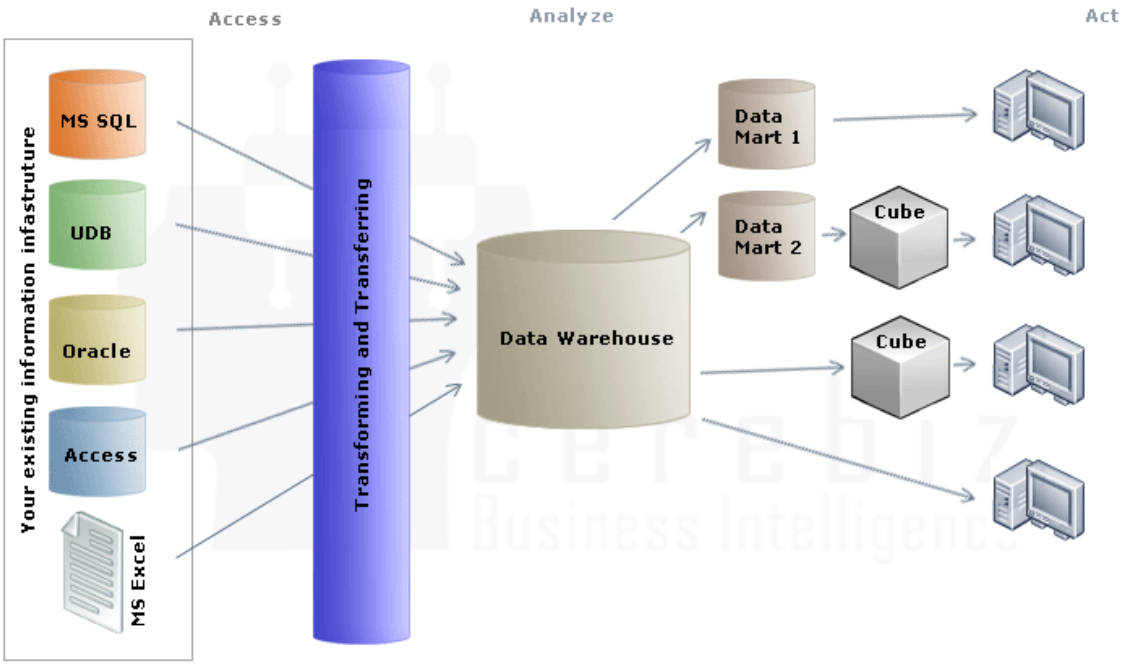
2.4.4 Konsep ETL

Proses yang penting untuk dilakukan dalam proses transformasi data adalah suatu proses pemindahan dari data operasional ke dalam *data warehouse* dengan melalui proses ETL (*Extract, Transform, Loading*) sehingga didapat data yang akurat dan sama. Dengan demikian data yang sudah melalui proses tersebut dapat digunakan sebagai sumber *data warehouse* yang dibangun.

Langkah-langkah transformasi data dalam instansi adalah:

- Membaca dan memilih data operasional yang berhubungan dengan informasi yang diminta, kemudian dilakukan penyaringan data yang akan digunakan atau tidak digunakan, lalu ditampung ke tempat penyimpanan data sementara.
- Melakukan penyeragaman data dan perubahan format data sebelum data masuk ke dalam *data warehouse* jika diperlukan.
- Melakukan transformasi dari tempat penyimpanan sementara ke dalam *data warehouse*.

Sumber data dari *data warehouse* berasal dari *database* operasional yang kemudian di konversikan ke dalam *data warehouse.* Fasilitas yang digunakan dalam proses transformasi yaitu menggunakan *SQL Server Integration Service* (SSIS) yang disediakan oleh *SQL Server 2005*



Gambar 2.9 Proses ETL ke *Data Warehouse*

(http://www.pervasiveintegration.com/dcontent/Collateral/DataWarehouse.pdf)